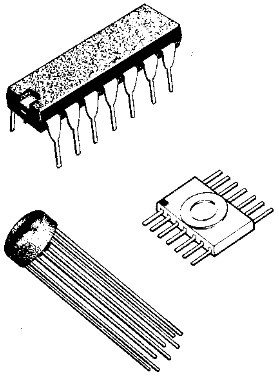
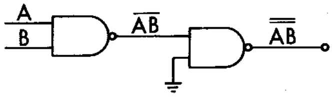
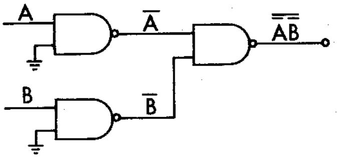
TEKTRONIX®

Digital Concepts

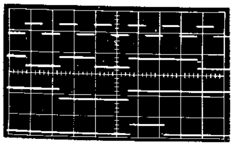






A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

$$X = \overline{\overline{AB}} = AB$$



A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

$$Y = \overline{\overline{A}\,\overline{B}} = A+B$$

OTHER BOOKS IN THIS SERIES:

Circuit Concepts

Power Supply Circuits
062-0888-01

Oscilloscope Cathode-Ray Tubes
062-0852-01

Storage Cathode-Ray Tubes and Circuits
062-0861-01

Television Waveform Processing Circuits
062-0955-00

Digital Concepts
062-1030-00

Spectrum Analyzer Circuits
062-1055-00

Oscilloscope Trigger Circuits
062-1056-00

Sweep Generator Circuits
062-1098-00

Measurement Concepts

Information Display Concepts
062-1005-00

Semiconductor Devices
062-1009-00

Television System Measurements
062-1064-00

Spectrum Analyzer Measurements
062-1070-00

Engine Analysis
062-1074-00

DIGITAL CONCEPTS

BY

LEONARD W. BELL

Significant Contributions

by

JOHN W. SHEPPARD

FIRST EDITION SECOND PRINTING JUNE 1969
062-1030-00
PRICE $1.00

TEKTRONIX, INC.; 1968
BEAVERTON, OREGON
ALL RIGHTS RESERVED

CONTENTS

FORWARD

This the first edition of the Tektronix Digital Concepts book uses Negative True Logic when explaining circuits throughout the book. This is due to the fact that most digital instruments designed at Tektronix prior to the publication of this book have had their circuits and logic diagrams explained in terms of Negative True Logic.

This is not meant to imply that Tektronix has standardized on Negative True Logic. There are times when Positive True Logic may be the more natural form to use.

The content of the book is as valid for explaining the concept of one system as for the other.

1

INTRODUCTION

Automated or programmed devices using integrated logic circuits (IC) become more common daily. The engineer or technician whose background lies largely with conventional or analog-type circuitry can have difficulty understanding digital diagrams filled with odd-shaped symbols. Most people with an electronics background are trained to use schematic diagrams which require consideration of each individual component and its contribution toward the operation of the circuit. In logic circuitry as implemented today, our point of interest is shifted upward an order of magnitude. Rather than considering each individual bit and piece, entire circuits are supplied in individual packages. It is not necessary to know the exact circuit configuration of the particular device because the device is encapsulated. Consider the Fairchild 914 NAND gate. Within the capsule are six transistors and numerous associated resistors. The only access we have to these transistors and resistors is through the eight pins. Therefore signal-tracing the circuitry within the IC (or chip as it is often called) is impossible. It is necessary to understand the relationship between the input and output signals, but no more. Since we cannot repair the 914 we can only replace it as a unit. This is universally true of presently available integrated circuitry.

This Digital Concepts book will help the beginner to approach digital instruments from the standpoint of circuit blocks rather than individual components. We begin by reviewing the concepts of the decimal and binary number systems. We next study the rules of Boolean algebra and its application to the field of digital logic circuitry. We then present the application of the algebra to the design, simplification, and understanding of these circuits.

To the designer, applications of Boolean algebra involve the basic design and simplification of a particular series of functions. He begins with a series of statements of what a circuit is to perform and implements these statements in a logic circuit. Determining a first approximation of the circuit, the designer next applies the principles of Boolean algebra to simplify. After simplification the resulting Boolean equations are translated again into circuitry. Frequently the second design is simpler and therefore less expensive.

The user of the completed instrument has other concerns. For him the circuits are already designed. His major problem is to interpret instrument operation from the diagrams supplied. He is required to understand the sometimes complicated interconnecting of the various IC's to determine the scheme of operation. This is particularly necessary in troubleshooting the complete instrument. Since most digital instruments available today were designed using the principles of Boolean algebra, diagrams supplied use logic symbols. To realize what the symbols mean and gain a finer appreciation for digital techniques, the technician must also be familiar with the basic principles of Boolean algebra. From the technician's standpoint, however, the methods of simplifying a device are of secondary importance.

This book concentrates on the interpretation of existing designs, although some of the principles that enter into completing the design are mentioned. Having considered basic principles of Boolean algebra and the basic symbology, we next procede to more complex designs. Finally, selected circuits taken from existing Tektronix digital instruments are analyzed. The book does not explain the overall operation of such instruments, but concentrates on those areas which are common, such as counters and registers.

A thorough study of the book should accelerate the student's understanding of digital instruments.

2

THE BINARY NUMBER SYSTEM

Digital instruments such as the digital voltmeter, the frequency counter, and the analog-to-digital converter may be broken down into hundreds (or thousands) of switching devices. A switch has two stable conditions, "on" and "off." When examining devices containing many switches, the decimal system is unhandy. Since the switch is a two-state device, a counting or numbering system based upon the value two is convenient. Such a numbering system is called the binary number system. Although unfamiliar to the average person, the binary number system is logical and easily learned.

binary number system

decimal number system

In the decimal number system ten symbols are used: 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9. A person counting paper clips, for example, and writing down the count, writes 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

For the tenth clip he has run out of symbols, therefore, he starts again with 0 and places a 1 to the left of the zero indicating that the count has reached 10 one time. The next count is 11, indicating 1 ten + 1 one = 11. When the count reaches twenty, note that the right-hand column begins with 0 again but this time a 2 is written to the left of 0. This indicates that the count has gone to ten a total of two times. The symbol 63 indicates 6 tens + 3 ones. Note that at the count 99, we have again exhausted the symbols so we repeat the change which occurred at ten and write 100 indicating 1 hundred + 0 tens + 0 ones.

Note that the change points are even powers of ten which are indicated $10^1 = 10$; $10^2 = 100$; $10^3 = 1,000$; $10^4 = 10,000$, etc. In a written number such as 10,349 we can determine the various powers of 10 which the number represents by the position of the written numbers, as $1 \times 10^4 + 0 \times 10^3 + 3 \times 10^2 + 4 \times 10^1 + 9 \times 10^0$.

In the binary numbering system only two symbols are used. Although the symbols are completely arbitrary we use the first two symbols of the Arabic numbering system in order to avoid having to memorize new symbols. To see how binary counting works let us again assume a person is to count paper clips and is to write the running total in binary form. He begins by writing 0 indicating that he has not counted yet. He counts the first clip and writes 1. He now has on his paper 0, 1. When he counts the second clip what does he do? In the binary system there are only two symbols, therefore, he resorts to the same method used in the decimal system, he writes a 0 and places a 1 to the left indicating he has counted to two 1 time. At the count of three he writes 11 indicating 1 two + 1 one = three. At the count of four he is again out of symbols so he writes 100 indicating 1 four + 0 twos + 0 ones. At the count of five he writes 101 indicating 1 four + 0 twos + 1 one and so he continues until at the count of seven he writes 111. Again he has used all symbols in all columns so he writes 1000, indicating 1 eight + 0 fours + 0 twos + 0 ones. Look at Fig. 2-1, which shows the binary count along with the same count in decimal form.

Note that the position notation idea is valid for a number in binary form, except that each position is based upon a power of two. For example, 20_{10} is 10100_2 (the subscripts are used to indicate the radix being used. The radix of a numbering system is simply the number of symbols that it uses.)

10100_2 = 1 sixteen + 0 eights + 1 four + 0 twos + 0 ones which could also be written: $10100_2 = 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0$. (any number to the zero power equals 1)

binary-to-decimal conversion

In the study of digital circuits it will be necessary sometimes to be able to convert a binary form number to decimal form. With the aid of a power-of-two chart this can be accomplished very easily.

Consider the number 1101. This can be read using the position value of each symbol as $1 \text{ x } 2^3 + 1 \text{ x } 2^2 + 0 \text{ x } 2^1 + 1 \text{ x } 2^0$. Referring to the table in Fig. 2-1:

$$\begin{array}{l} 1 \text{ x } 2^3 = 8 \\ 1 \text{ x } 2^2 = 4 \\ 0 \text{ x } 2^1 = 0 \\ \underline{1 \text{ x } 2^0 = 1} \\ \qquad\qquad 13_{10} \end{array}$$

is the number in decimal form.

DECIMAL	BINARY
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111
16	10000
17	10001
18	10010
19	10011
20	10100
21	10101
22	10110
23	10111
24	11000
25	11001
26	11010
27	11011
28	11100
29	11101
30	11110
31	11111
32	100000

Fig. 2-1. Comparison of binary and decimal numbers.

Comparing the same number in binary and decimal forms shows that the binary form is cumbersome in that it takes many more digits to express a number. Refer back to Fig. 2-1, and notice that 32_{10} takes 6 digits in binary form. Why then do digital instruments use the binary system? Electronic devices and decimal counting are not very compatible. Although circuits can be built to use base-10 values, the circuitry is quite complex and involves the use of ten different voltage levels.

Since active electronic devices can operate as switche two-voltage-level circuits are easily made. In addition these devices can be made to switch at rates of millions per second. It is simplicity and speed which makes the use of the binary system practical in electronics.

bit

word

The operation or programming of digital instruments often requires that very long binary numbers be used. For convenience, certain terms are used to identify parts of these numbers. The term *bit* is used to identify a binary digit. (Bit is derived from BInary digiT.) The term *character* is a group of bits. The term *word* refers to the total number of bits required by a particular instrument.

character

For example, the Tektronix Type 240 Program Control Unit is designed to process a binary number which is 192 bits long. The complete number is called a word and the Type 240 is said to use a 192-bit word. Because of the extreme length of the word, for convenience it is divided into groups of 4 bits. Each 4-bit group is called a character. Hence the Type 240 is also said to use a 48-character word.

binary codes

Digital instruments use data and instructions in binary form. Humans, however, use decimal numbers and alphabetic letters. Therefore, various codes have been designed to facilitate communication with digital devices. These codes are formed by taking groups of bits and assigning each unique combination a particular letter, symbol or decimal number. There are many codes in existence, only a few of which will be considered here.

Some binary codes use a number weighting scheme. The simplest code called pure binary uses the exact position value of each binary digit as the weight

DECIMAL	BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Fig. 2-2. 8, 4, 2, 1 BCD code.

value. For example, the number 15_{10} is written in binary as 1111. This number is read as 1 x 8 + 1 x 4 + 1 x 2 + 1 x 1 = 15. Pure binary (also called hexadecimal) is said to have an 8, 4, 2, 1 weight. Many other weight schemes are used. Examples include 7, 4, 2, 1; 4, 2, 2', 1 and 6, 3, 2, 1, 0 (5 bits).

Other codes are unweighted which means that the decimal equivalent of the binary number is determined only by an arbitrarily assigned value. An example of this type is the Excess-3 code.

binary-coded decimal

The simplest code to understand is the binary-coded decimal, which is abbreviated BCD. The BCD code uses four binary bits per character and a weight scheme of 8, 4, 2, 1. Each character has the decimal value that the four bits represent. The code is shown in Fig. 2-2. Note that the decimal equivalent is simply the binary number expressed in decimal form.

A 4-bit number can have values from zero to fifteen. Ordinarily, however, in the BCD code only enough combinations are used to express all 10 decimal symbols. In order to express decimal numbers greater than 9, a separate four-bit group is used for each number. For example: 82_{10} is 1000 0010 in BCD, 370_{10} is 0011 0111 0000, 591_{10} is 0101 1001 0001.

Note that the BCD system requires many bits to express a decimal number.

To return to the Tektronix Type 240 Program Control Unit, recall that the 192-bit word is divided into 4-bit characters. Each 4-bit character is further simplified by giving each character its decimal value in a specific case. Since each 4-bit character in this situation may contain any of the sixteen possible combinations of bits, a character in the Type 240 may have a value in excess of nine. Fig. 2-3 shows all possible values.

DECIMAL	CHARACTER
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

Fig. 2-3. Pure binary 8, 4, 2, 1.

DECIMAL	8,4,2,1 CODE	4,2,2',1 CODE	EXCESS-3 CODE
0	0000	0000	0011
1	0001	0001	0100
2	0010	0010	0101
3	0011	0011	0110
4	0100	1000	0111
5	0101	0111	1000
6	0110	1100	1001
7	0111	1101	1010
8	1000	1110	1011
9	1001	1111	1100

Fig. 2-4. Comparison of some BCD codes.

This coding is similar to BCD but includes combinations which are forbidden in the BCD system. To reduce confusion, care should be taken not to call the Type 240 character system "BCD." It should instead be called pure binary 8, 4, 2, 1.

4,2,2',1 code

Excess-3 code

Other common codes are shown in Fig. 2-4; the 8, 4, 2, 1 BCD code is included for comparison. The 4, 2, 2', 1 code is used in the Tektronix Type 6R1A. The Excess-3 code is formed by adding binary 3 to the BCD number. For example, 0_{10} in BCD is 0000; by adding 3_2 the sum is 0000 + 0011 = 0011. Each Excess-3 number is formed by the same process. The Excess-3 code has some advantages over BCD when performing arithmetic subtraction in computers.[1]

[1] T.C. Bartee, *Digital Computer Fundamentals*, (New York: McGraw-Hill, 1966), pp 56-7.

In computers designed for business data processing it is necessary to work with alphabetic characters as well as decimal numeric characters. Such an alphanumeric code must contain more than 4 bits since 26 letters plus 10 digits must be encoded. This means that at least 6 bits must be used since 5 bits contain only 32 unique combinations. A six-bit code has often been used. In the past each manufacturer has selected or created codes to suit his particular devices. In an attempt to standardize, the American Standards Association approved a new 7-bit code in 1964. This code is known as ASCII (American Standard Code for Information Interchange). (For verbal communication the letters are phoneticized az-key.) Fig. 2-5 shows the entire code. Seven bits are used so that punctuation marks, symbols, plus telephone and teletype abbreviations can be included.

ASCII Code

Examine the column headed by "011." The ten decimal digits are listed in order. The chart is decoded by using the four digits shown on the left and adding the three digits at the head of the column. Examples: 4 = 011 0100 and 7 = 011 0111. The last four binary digits express the decimal number in 8, 4, 2, 1 BCD code. Because of this, the ASCII code is compatible with instruments designed to use the 8, 4, 2, 1 BCD code. The Type 240 Program Control Unit can be addressed by the ASCII code.

Standard Code

b7 b6 b5 Bits					0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b4	b3	b2	b1	COLUMN / ROW	0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	´	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	¦
1	1	0	1	13	CR	GS	–	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

Legend

Control Characters

NUL	Null
SOH	Start of Heading (CC)
STX	Start of Text (CC)
ETX	End of Text (CC)
EOT	End of Transmission (CC)
ENQ	Enquiry (CC)
ACK	Acknowledge (CC)
BEL	Bell (audible or attention signal)
BS	Backspace (FE)
HT	Horizontal Tabulation (punched card skip)(FE)
LF	Line Feed (FE)
VT	Vertical Tabulation (FE)
FF	Form Feed (FE)
CR	Carriage Return (FE)
SO	Shift Out
SI	Shift In
DLE	Data Link Escape (CC)
DC1	Device Control 1
DC2	Device Control 2
DC3	Device Control 3
DC4	Device Control 4 (Stop)
NAK	Negative Acknowledge (CC)
SYN	Synchronous Idle (CC)
ETB	End of Transmission Block (CC)
CAN	Cancel
EM	End of Medium
SUB	Substitute
ESC	Escape
FS	File Separator (IS)
GS	Group Separator (IS)
RS	Record Separator (IS)
US	Unit Separator (IS)
DEL	Delete

NOTE: (CC) Communication Control
(FE) Format Effector
(IS) Information Separator

Graphic Characters

Column/Row	Symbol	Name
2/0	SP	Space (Normally Non-Printing)
2/1	!	Exclamation Point
2/2	"	Quotation Marks (Diaeresis)
2/3	#	Number Sign
2/4	$	Dollar Sign
2/5	%	Percent
2/6	&	Ampersand
2/7	´	Apostrophe (Closing Single Quotation Mark; Acute Accent)
2/8	(	Opening Parenthesis
2/9	)	Closing Parenthesis
2/10	*	Asterisk
2/11	+	Plus
2/12	,	Comma (Cedilla)
2/13	–	Hyphen (Minus)
2/14	.	Period (Decimal Point)
2/15	/	Slant
3/10	:	Colon
3/11	;	Semicolon
3/12	<	Less Than
3/13	=	Equals
3/14	>	Greater Than
3/15	?	Question Mark
4/0	@	Commercial At
5/11	[	Opening Bracket
5/12	\	Reverse Slant
5/13	]	Closing Bracket
5/14	^	Circumflex
5/15	_	Underline
6/0	`	Grave Accent (Opening Single Quotation Mark)
7/11	{	Opening Brace
7/12	¦	Vertical Line
7/13	}	Closing Brace
7/14	~	Overline (Tilde ; General Accent)

Fig. 2-5. USA Standard Code for Information Interchange.

octal system

Another common numbering system used within the digital area is the octal system. The octal system is based on the number 8. Eight digits are used, 0, 1, 2, 3, 4, 5, 6, and 7. The rules are basically the same as for binary or decimal except that position is based upon powers of 8. Fig. 2-6 shows decimal and octal equivalents.

Note that the octal system requires more digits than the decimal system to express a number but not nearly as many as the binary system. The octal system converts readily to binary because the basis of the octal system 8 is also an even power of two, i.e., $8 = 2^3$.

Caution must be used in verbally naming numbers expressed in octal and other numbering systems. For example, 10_8 is *not* pronounced ten because $10_8 = 8$ and should be called "eight" verbally.

The octal numbering system is used by several digital-equipment manufacturers as a means of expressing binary numbers by using fewer symbols. This system could be called "octal-coded binary." For example, the Digital Equipment Corporation makes the PDP-8 family of computers. These computers operate with a 12-bit word. A word might be 110 011 001 111. To reproduce this word would of course require writing 12 digits. By arranging the word bits in groups of three bits each, and converting each group to its equivalent in octal code, the same number can be written using 4 octal digits. The process is shown in Fig. 2-7.

Thus the 12-bit word 110 011 001 111 can be written 6317_8. This system is convenient because a group of 3 bits can have only 8 possible values. With practice the numbers from 000_2 to 111_2 can be memorized and the binary-to-octal conversion can be performed mentally.

This is primarily used as a shorthand method of writing binary numbers. A computer program might consist of several hundred 12-bit words, each one of which must be recorded. Think how much writing can be saved by using the octal-coded binary method of condensing the binary word! Seldom will the octal numbering system be used for arithmetic operation; it is the positional notation which is of value here.

DECIMAL	OCTAL
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	10
9	11
10	12
11	13
12	14
13	15
14	16
15	17
16	20
17	21
18	22
.	.
.	.
.	.
64	100
65	101
.	.
.	.
.	.

Fig. 2-6. Octal numbering system.

OCTAL	6	3	1	7
BINARY	110	011	001	111

Fig. 2-7. Octal-to-binary conversion.

3

BOOLEAN ALGEBRA

The engineer's understanding of digital circuits and digital instruments requires an understanding of a different form of algebra from the algebra taught in high school. Although unfamiliar to many, this algebra is logical and easily understood. Boolean algebra, universally used by digital instrument designers, differs from conventional algebra in that it uses the binary numbering system. Boolean algebra contains methods which are specially adaptable to digital circuitry and makes the design of such circuitry much easier. Conventional algebra is best for everyday use, but in the digital area, it may needlessly complicate circuit design.

There is a twofold advantage in using Boolean algebra in the digital field. First, Boolean algebra permits the engineer to design a circuit or instrument in a logical manner. Secondly, it allows another engineer or technician to easily understand and follow the operation of the device.

two-valued logic

Boolean algebra has been called the algebra of two-valued logic. An English mathematician, George Boole, published a work in 1854 titled, *An Investigation of the Laws of Thought*. This book contains one of the earliest attempts to discuss logic in a mathematical sense using special notation similar to mathematical symbols.

Boolean algebra remained almost forgotten until 1938 when Claude Shannon, a research assistant at MIT, published a thesis titled, "A Symbolic Analysis of Relay and Switching Circuits." The paper presented a method for representing switching circuitry by a set of mathematical expressions analogous to the expressions of Boolean algebra. The techniques developed in Shannon's paper have been improved until today they are used in all parts of digital circuit design. The economy of reducing circuitry to mathematical expressions and simplifying by mathematical operations permits the design of even the most complex modern computers.

Fig. 3-1.

Boolean algebra is a method of manipulating deductive logic. It recognizes only two possible values for a statement. A statement is either entirely *true* or entirely *false*. There are no halfway conditions. A statement which is not true must therefore be false. These premises allow the algebra to be used to represent the conditions found in electrical switching circuitry. Consider the switch of Fig. 3-1. The switch is either open or closed. It has no other possible conditions. By applying the basic premise of Boolean algebra we can define the closed switch as a "true" condition and the open switch as a "false" condition. The switch, when not open, must be closed. If not closed it is open. This parallels the Boolean logic. The closed condition could be called the false state and the open condition a true state, without ambiguity. By defining the conditions of a two-state device in Boolean terms, the symbology of the algebra becomes usable. Note the switch of Fig. 3-1 can also itself represent various electronic elements such as transistors, diodes, and vacuum tubes operated in switched modes.

symbology

Boolean algebra has numerous theorems; however, only a few need be examined here. Letters are used to represent quantities. Letters close to the beginning of the alphabet are used to represent variable values and letters close to the end of the alphabet represent unknown quantities.

NOT

Consider the circuit of Fig. 3-2. We represent the condition of switch SWI by the letter A if the switch is closed and by $\bar{A}$ if the switch is open. The bar over A indicates the "false" state of the switch where false indicates the open condition. An expression with the bar is read aloud by saying for $\bar{A}$, "not A." It follows therefore, that the expression A indicates, by the absence of the bar, that the switch is in a closed or true state. Throughout Boolean algebra this convention is followed.

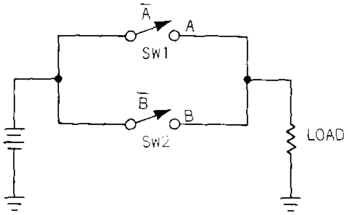


Fig. 3-2.

inverter

The not symbol is used to represent a very common circuit function which is the inverter. A signal can be inverted by a transformer, amplifier, or by various other means. Assume a two-valued signal is assigned the letter F. If the signal is inverted by a circuit the value of F must also be inverted. Symbolically $\bar{F}$ is placed at the output. The symbol for an inverter presently used by Tektronix on logic diagrams is shown in Fig. 3-3.

Considering again Fig. 3-2, we can represent the condition of switch SW2 with the letter B. Further, we can describe the presence or absence of voltage across the load by the letters T (true) or F (false). Thus, T means voltage is present and F means voltage is not present. A Boolean algebra equation may now be written which describes all possible combinations of the switches SW1 and SW2 and whether or not voltage appears across the load. The equation is $A + B = X$, where X represents a voltage across the load.

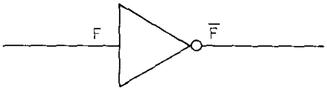


Fig. 3-3. Inverter symbol.

A	B	X
$\bar{A}$	$\bar{B}$	$\bar{X}$
$\bar{A}$	B	X
A	$\bar{B}$	X
A	B	X

(A)

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

(B)

Fig. 3-4. OR function truth tables.

OR

Reading aloud the reader should state, "A OR B equals X." The symbol "+" is read as "OR" in Boolean algebra, not as "plus." Literally, the equation states, "If either A OR B (or both) is true, then X is true." To properly examine this equation requires the use of a table listing all possible combinations of A and B. Such a table is easily constructed. See Fig. 3-4A.

truth table

This table shows that there are four possible combinations for the variables A and B. It follows that in a different equation with three variables there would be eight combinations, and with four variables sixteen combinations, i.e., the number of combinations for 2-valued variables is 2^n, where n is the number of different variables. Note that X is true for all conditions of A and B except where A and B are both false (switches SW1 and SW2 open). The truth table is easier to construct and interpret by using the binary number symbols 1 and 0, as in Fig. 3-4B. In this table wherever a variable is *true* a 1 is placed, wherever a variable is *false*, a 0 is placed.

The above discussion describes a Boolean OR function. For the purpose of simplifying diagrams wherever a circuit appears which could perform an OR function, the schematic may be replaced by the OR-gate symbol. The presently used symbol is shown in Fig. 3-5A. Three variables are shown. The distinctive shape means the OR function. Fig. 3-5B shows the Boolean equation for the OR gate.

One of the many possible circuits which operates as an OR gate is shown in Fig. 3-5C. To realize how this circuit functions, let us define two voltage levels. A voltage level of +10 V is defined as a logical 0 in this circuit. A ground level (0 V) is a logical 1.

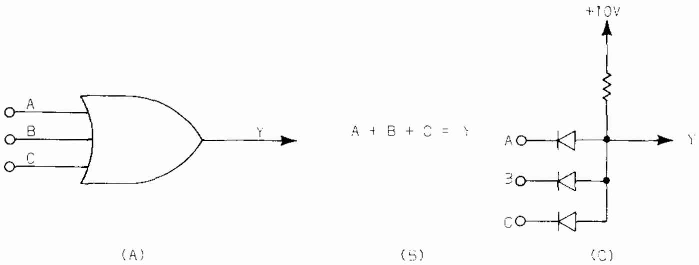


Fig. 3-5. OR functions.

negative logic

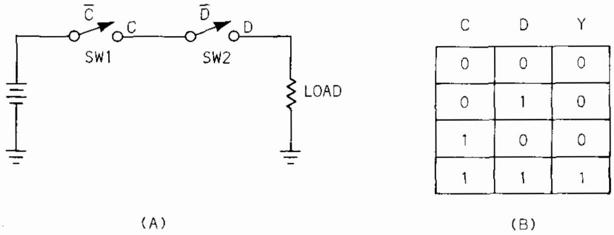
The absolute voltage level assigned to logical 1 is negative with respect to the voltage level assigned to logical zero. This is an example of negative logic. The exact voltage levels could be any values desired, but if the more negative of the two represents 1 then the logic is negative.

The reverse case is possible and often used. If, for example, +10 V was logical 1 and zero volts logical 0 the circuit would be termed a "positive logic" circuit. Note, however, that the circuit of Fig. 3-5C is *not* a positive logic OR gate. Study of the circuit shows that if A or B or C were at 0 V then the output Y would also be at 0 V, and therefore logically true (1) as defined above.

AND

Another important Boolean algebra function is called the AND function. Fig. 3-6 illustrates a circuit which will help one understand the concept of the AND function.

Note that unless switch SW1 *and* SW2 are closed, no voltage is delivered to the load. If we assign symbols, letting C stand for switch SW1, D stand for switch SW2, and Y stand for presence of output across the load, then a Boolean equation for the circuit may be written C • D = Y. Note the use of the symbol which means "multiply by" in conventional algebra. In Boolean the "•" means "AND."



C	D	Y
0	0	0
0	1	0
1	0	0
1	1	1

Fig. 3-6. AND function.

The equation is read aloud as "if C is true AND D is true, then Y is true." Often an expression omits the dot entirely but its presence is understood; for example, BEFGH would mean B AND E AND F AND G AND H.

Fig. 3-6B illustrates the truth table for the circuit of Fig. 3-6A. Note that the only time the output is true is when the inputs are all true.

Fig. 3-7A shows the symbol for an AND gate as found on logic diagrams. Fig. 3-7B shows the equation for the gate and Fig. 3-7C shows the truth table for the example. Note again that Y is true only when A and B and C are all true.

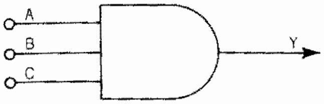
An example of a circuit which displays the AND function is shown in Fig. 3-8. This is an AND gate for negative logic. If any one of the inputs A, B, or C is false (at +10 V), output Y will be false. Only if A•B•C are true (0 V) will Y be true.

Fig. 3-5C and 3-8 are examples of OR and AND gates. There are numerous other ways of building circuits and devices to perform these functions. The principle methods will be treated in a later chapter.

positive logic

As an experiment let us re-examine the circuit of Fig. 3-8, but this time with positive logic. We define +10 V as the true level and 0 V as the false level. If +10 V is now the true level then Y will be at +10 V if A or B or C (or any combination of A, B, C) is at 10 V. The truth table inverts the values of Fig. 3-7C.

Constructing a new truth table as in Fig. 3-9, it is apparent the circuit is now an OR gate. Thus, a negative logic AND gate is a positive logic OR gate. At this time, examine Fig. 3-5C and you will find that the circuit is a positive logic AND gate. Thus, depending on the logic chosen for a particular device, the gates are dual in nature. Notice that once the logic levels are chosen this duality vanishes.



(A)

$A \cdot B \cdot C = Y$

(B)

A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

(C)

Fig. 3-7. Three-variable AND function.

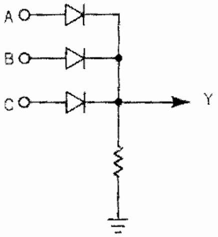


Fig. 3-8. AND gate.

A	B	C	Y
1	1	1	1
1	1	0	1
1	0	1	1
1	0	0	1
0	1	1	1
0	1	0	1
0	0	1	1
0	0	0	0

Fig. 3-9. OR-function truth table.

$A + A = A$	3.1
$A \cdot A = A$	3.2
$A + 1 = 1$	3.3
$A \cdot 1 = A$	3.4
$A + 0 = A$	3.5
$A \cdot 0 = 0$	3.6
$A + \overline{A} = 1$	3.7
$A \cdot \overline{A} = 0$	3.8
$\overline{\overline{A}} = A$	3.9
$A + AB = A$	3.10
$A\ (A + B) = A$	3.11
$\overline{A + B} = \overline{A}\ \overline{B}$	3.12
$\overline{AB} = \overline{A} + \overline{B}$	3.13
$A\ (\overline{A} + B) = AB$	3.14
$A + \overline{A}B = A + B$	3.15
$\overline{A} + AB = \overline{A} + B$	3.16
$\overline{A} + A\overline{B} = \overline{A} + \overline{B}$	3.17

Table 3-1. Useful Boolean algebra theorems.

Boolean theorems

Boolean algebra is the tool which enables the engineer to reduce circuitry to mathematical equations and then to simplify these equations. Having studied some of the Boolean algebra functions and examined the circuit implementation of these functions, we next consider some of the theorems and postulates of Boolean algebra. Some of these postulates are exactly the same as in ordinary algebra. Some, however, are exclusive to Boolean algebra. Table 3-1 shows a list of some useful Boolean algebra theorems. Some of the theorems can be seen to be true by inspection, some however, require proof. We shall prove several of the theorems.

The proof may be accomplished by several different methods. One method uses the truth table, another proves the theorem in mathematical form by applying previously proven algebraic theorems to simplify the mathematical equations. A third method implements the Boolean function in an actual circuit and is simplified by inspection. We shall give examples of all of these methods. Fig. 3-10A shows a circuit which implements Theorem 3.1: A OR A = A. Since the quantity A is to be OR'd with itself, A is represented as a ganged switch. Whenever the switch is closed, we can assume A is true, whenever the switch is open, we can assume it is false. The extra contact on the switch is redundant.

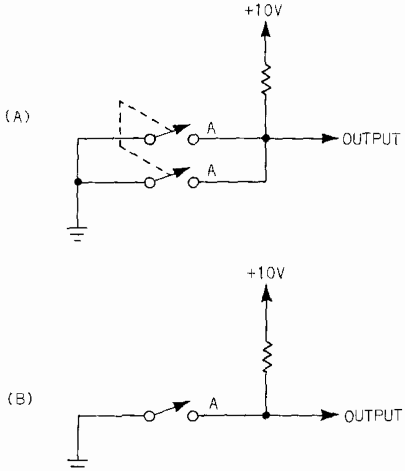
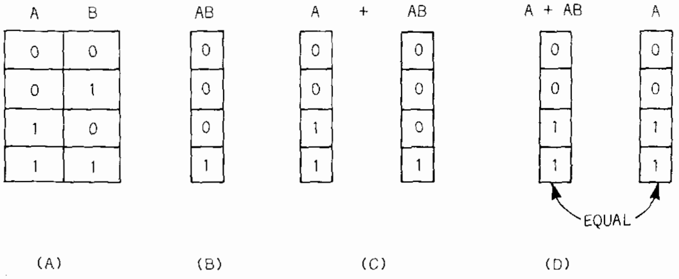


Fig. 3-10.

Therefore, the expression and the circuit could be simplified to a single switch labeled A as in Fig. 3-10B.

Theorem 3.10 will be proven by the use of a truth table. See Fig. 3-11. Since the theorem involves two variables, A and B, the figure lists all four combinations of values that A and B can assume. The values for the term AB are shown in Fig. 3-11B. In Fig. 3-11C we combine the values of A OR'd with the quantity of A AND B. In Fig. 3-11D we find the value of the expression A OR the quantity A AND B is exactly the same as the value of A alone. Therefore, when an expression of the form A + (A·B) appears in an equation, it can be replaced solely by the quantity A.



A	B	AB	A	+	AB	A + AB	A
0	0	0	0		0	0	0
0	1	0	0		0	0	0
1	0	0	1		0	1	1
1	1	1	1		1	1	1

Fig. 3-11

Theorem 3.11 is proved mathematically:

A (A + B)	=	A	Theorem 3.11
AA + AB	=	A	Multiply
A + AB	=	A	By 3.2
A	=	A	By 3.10

DeMorgan's theorems

The proof of the rest of the theorems is left as an exercise for the reader. Pay special attention to theorems 3.12 and 3.13 which are known as DeMorgan's theorems. These theorems form the basis for NAND and NOR operations described in Chapter 4.

The engineer may be presented with digital circuit problems in several different forms. One, he may be given a series of logical statements which may be translated into actual circuitry. Two, he may desire to implement a truth table by actual circuitry. Three, he may be given a logic diagram representing a Boolean algebra function and, four, he may be presented with the expression in mathematical terms. In all cases he should be aware of the methods by which one form can be changed to any other. As an example, supposing the problem is to implement the following logical statement. "A room with two doors is to have a central light installed with switches accessible to each door, either one of which can turn the light either on or off." Let the letter A represent the switch by one door, and the B represent the switch by the other door. Let the letter L stand for the lamp. We first construct the truth table. See Fig. 3-12. Although the choice is entirely arbitrary we assume that when a switch is closed it has a logical value of 1 and when opened it has the logical value of 0.

A	B	L
0	0	0
0	1	1
1	0	1
1	1	0

Fig. 3-12. Exclusive-OR truth table.

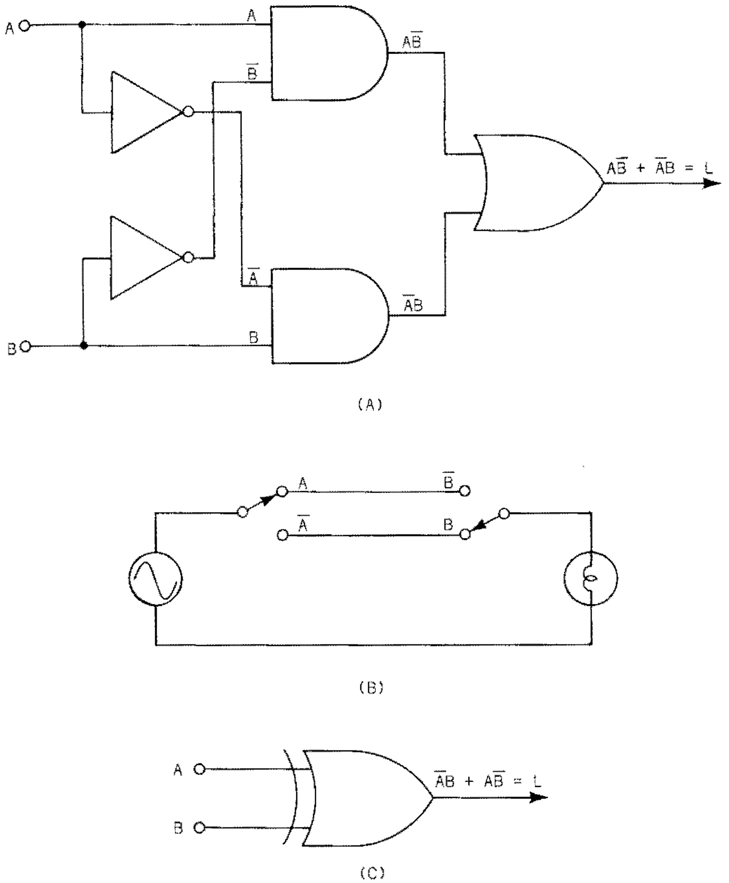
When A is zero and B is zero the lamp will be off -- we assign this condition a logical value of zero. In order for either switch to control the lamp, if A or B changes states, the lamp must go on. Therefore, when A is zero and B is one L must have the value of one and when A is one and B is zero, L must also have a value of one. Finally, when A and B are both one the lamp must be off. The next problem is to write a Boolean algebra expression for the truth table.

L is to be one for two possible conditions; when A is zero and B is one and also when A is one and B is zero. When two quantities are AND'd together, they must be both equal to one for the result to be one. (Theorem 3.2). In this case we indicate A as zero, however, its complement $\bar{A}$ would have a value of one when A is zero. Therefore, we write: $\bar{A}B$. Similarly we write $A\bar{B}$. L is 1 for either one OR the other combination. The complete equation is $\bar{A}B + A\bar{B} = L$.

Inspection of the equation shows that it cannot be simplified using any of the theorems in Table 3-1. Next, we implement the equation in a logic diagram. Wherever two terms are ANDed together we use the symbol for an AND gate; wherever the two terms are OR'd together, we use the symbol for an OR gate. The complete diagram is shown in Fig. 3-13A. Inverters are used to generate the negated values of any one of the variables when it is required.

In Fig. 3-13B we show the final circuit which uses switches to implement the logic functions. By the use of single-pole double-throw switches we generate both the true and negated value for a particular variable.

Re-examining Fig. 3-13A, note that L is true if either A or B is true, but not when both are true. This particular combination is so useful that it has been given a special name and a special symbol. The implementing circuit is called an "Exclusive OR" gate. The symbol is shown in Fig. 3-13C which diagrams the lamp problem using the Exclusive OR. The problem could be diagrammed as in Fig. 3-13A or 3-13C; however, the latter diagram is the more convenient.



(A)

(B)

(C)

Fig. 3-13. Exclusive-OR operations.

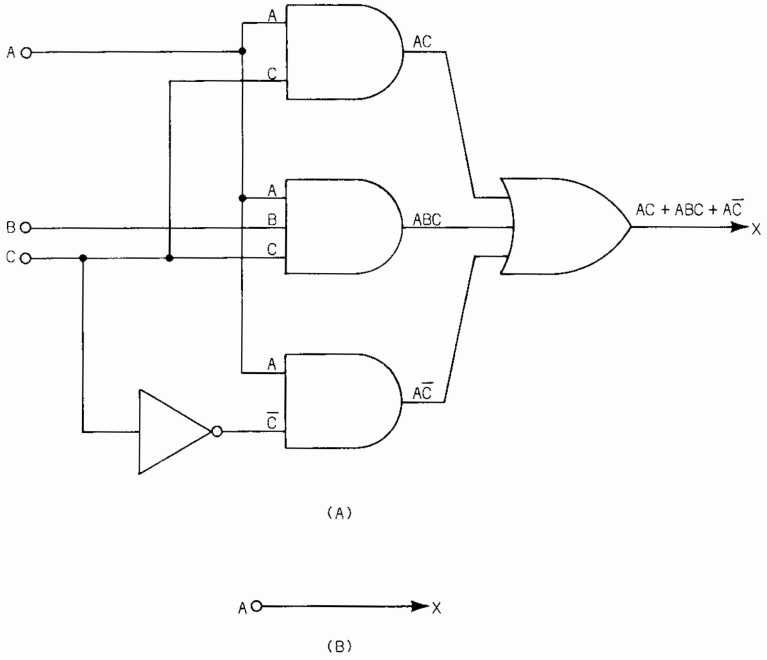


Fig. 3-14. Simplification of complex logic diagrams.

Fig. 3-14A shows another type of problem that may be encountered. The engineer is presented with an accomplished logic diagram. His task is to reduce this diagram to its Boolean algebraic equivalent. By starting at the input and carefully labeling each line, noting also where negation or inversion has taken place, the function may be completely derived from the diagram. The final equation is shown at the output. Examining the equation shows that the equation can be simplified as follows:

$AC + ABC + A\overline{C} = X$

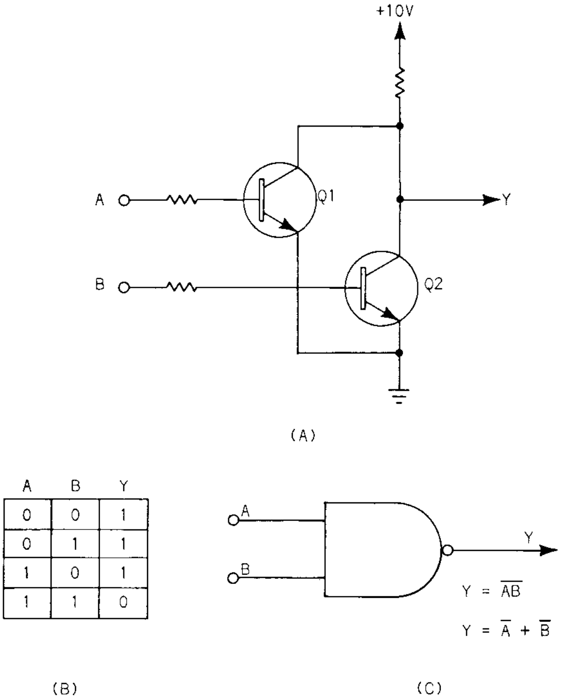
$A(C + \overline{C}) + ABC = X$ Factoring and rearranging terms.

$A(1) + ABC = X$ By 3.7

$A + ABC = X$ By 3.4

$A = X$ By 3.10

The simplified equation shows that the function reduces to a straight-wire connection (Fig. 3-14B) from A to X eliminating all other gates and connections.



A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Fig. 4-1. NAND functions.

4

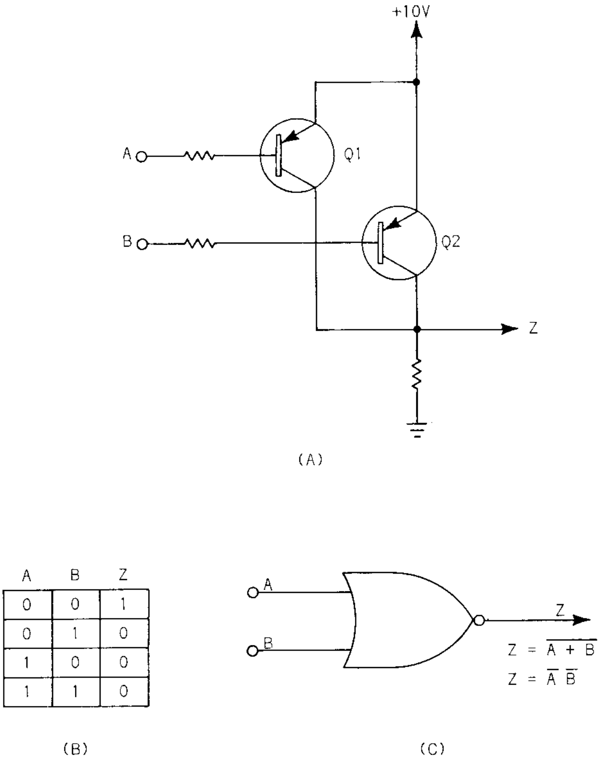
NAND GATE, NOR GATE, AND FLIPFLOP

When the design engineer tries to implement a Boolean algebra function he may use any common switching device, including the transistor and vacuum tube. When operated in common-emitter mode, the transistor acts as an inverter. Thus, when a transistor is used to implement a logic function; the output of the transistor, if taken at the collector, represents the inversion of the input operation. This means that when constructing an AND gate using a transistor, often the output represents the inversion of an AND gate.

In Fig. 4-1A we show a two-input digital gate, consisting of two NPN transistors Q1 and Q2. In a negative logic system, when inputs A and B are true they are 0 V. In this condition both transistors are off. If they are both off, the output level is +10 V, the false level. Examination of the truth table for an AND gate shows that the output is 0 or false for any combination of inputs A and B except when *both* are true. Referring to Fig. 4-1A again, if both transistors are off, which occurs only when A and B are true, Y is false. For all other conditions, one or the other or both of the transistors is on because its base is at the false or positive level. With either transistor on, Y is equal to 0 V, the true level. The Y column is the inverse of an AND gate output.

NAND

This type of gate is known as a negated AND gate which is shortened to NAND gate. The symbol for a NAND gate appears in Fig. 4-1C. The basic shape of the gate identifies it as an AND function. The circle at the output of the gate means that the signal is logically *inverted* at the point where the circle appears. The Boolean algebra expression for the NAND GATE is written as $Y = \overline{AB}$. By DeMorgan's theorem (3.13) this expression can also be written as $Y = \overline{A} + \overline{B}$. Note that in one form of the equation, the AND function is indicated, in the other form of the equation the OR function is indicated.



A	B	Z
0	0	1
0	1	0
1	0	0
1	1	0

Fig. 4-2. NOR functions.

NOR

Fig. 4-2A shows a digital logic circuit using two PNP transistors. In negative logic system when inputs A or B are made true, the base of the appropriate transistor is pulled negative with respect to the emitter which turns the transistor on. A truth table for this gate is shown in Fig. 4-2B. Output Z is true only when both transistors are not conducting. Both transistors are off only when A and B are both false. This result is equivalent to taking the output of an OR gate and negating it. The circuit is therefore referred to as a negated OR gate or NOR gate. The symbol for a NOR gate is shown in Fig. 4-2C. The basic shape of the gate indicates an OR function.

The presence of a circle at the output indicates logic inversion at that point. An equation for a NOR gate is $Z = \overline{A + B}$. By DeMorgan's Theorem (3.12) this expression is also equivalent to $Z = \bar{A}\,\bar{B}$. Thus, in the NOR gate as in the NAND gate both OR and AND functions can be implemented. Neither the NAND gate nor the NOR gate are restricted to two input configurations. NAND or NOR gate IC's are available with up to five inputs.

NAND or NOR as AND or OR

The question is frequently asked as to why most commercially available digital integrated-circuit gates are of the NAND or NOR variety. The answer is found by applying DeMorgan's theorem to NAND and NOR functions. As mentioned previously DeMorgan's theorem shows that in either the NOR or the NAND gate, both AND and OR functions are indicated.

A manufacturer producing integrated-circuit chips can manufacture a single type of gate, either NAND or NOR type, thus simplifying his own inventory and production problems. The user of this single type of gate can implement any kind of indicated operation AND/OR strictly by the use of NAND or NOR gates. To illustrate this principle let us re-examine Theorem 3.12 which reads: $\overline{A + B} = \bar{A}\,\bar{B}$. See Fig. 4-3. The output of the gate is $\overline{\bar{A} + \bar{B}}$ which equals $\bar{\bar{A}}\,\bar{\bar{B}}$. By applying Theorem 3.9, $\bar{\bar{A}}\,\bar{\bar{B}}$ = AB. By inverting logic levels before the NOR gate, the result is equivalent to an AND function.

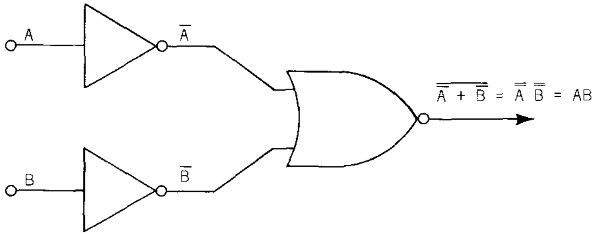
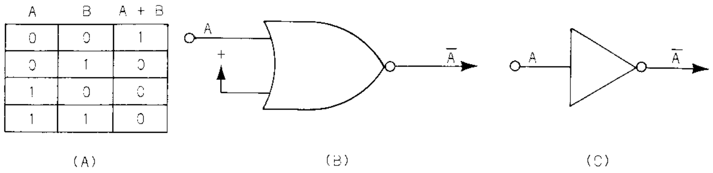

A
$\bar{A}$
B
$\bar{B}$
$\overline{\bar{A} + \bar{B}} = \bar{\bar{A}}\,\bar{\bar{B}} = AB$

Fig. 4-3. Implementing the AND function with NOR gates.



A	B	A + B
0	0	1
0	1	0
1	0	0
1	1	0

Fig. 4-4. The NOR gate as an inverter.

An important question is: Can the NOR gate function as an inverter? Examining the truth table for a NOR gate Fig. 4-4A, if input B is held permanently false, when A is 0 the output is 1 and when A is 1 the output is 0. This method of operating a NOR gate would invert whatever signal is connected to input A. Input B is easily held false in a negative logic system by connection to the positive supply. A NOR gate operated as an inverter, Fig. 4-4B, could also be symbolized by the symbol in Fig. 4-4C, i.e., an inverter or NOT circuit. It is of little consequence in a logic diagram exactly how the inverter function is implemented. Many diagrams would use the inverter symbol. The AND function of Fig. 4-3 could therefore be implemented by using three NOR gates.

If the OR function is to be implemented using NOR gates, the designer may proceed as in Fig. 4-5. $\overline{A + B}$ inverted becomes A + B; thus inversion of the AND function, or the OR function, can be implemented using nothing more than NOR gates.

Theorem 3.13 reads $\overline{AB} = \overline{A} + \overline{B}$. Using the A NAND B relationship any of the desired functions can be implemented. Examine the truth table for a NAND gate as shown in Fig. 4-6A. If the B input is held permanently true (Fig. 4-6B) then the A input is inverted. The NAND gate may be used as an inverter and may be symbolized as in Fig. 4-6C.

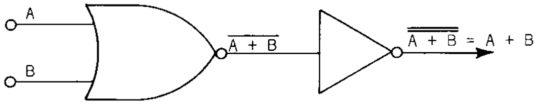
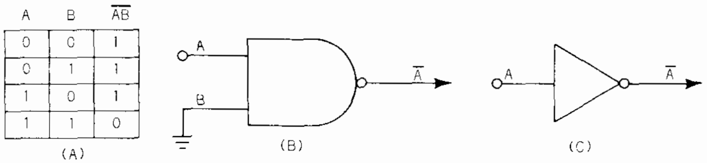


Fig. 4-5. Implementing the OR function with NOR gates.

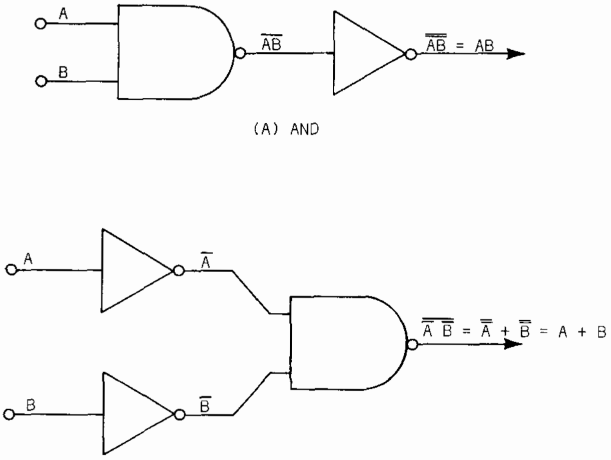


A	B	$\overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

Fig. 4-6. Inverting with a NAND gate.

Fig. 4-7A illustrates the implementation of an AND function using only NAND gates. Fig. 4-7B shows the implementation of an OR function using only NAND gates. A and B are first inverted and then NANDed together giving the result, $\overline{\overline{A}\ \overline{B}}$. By DeMorgan's theorem this is equivalent to A + B.

Most integrated circuit chips presently manufactured are NAND gates when used in a negative logic system. Very many NAND gates appear in Tektronix digital instruments. The student should be familiar with the implementation of logic functions using NAND gates.

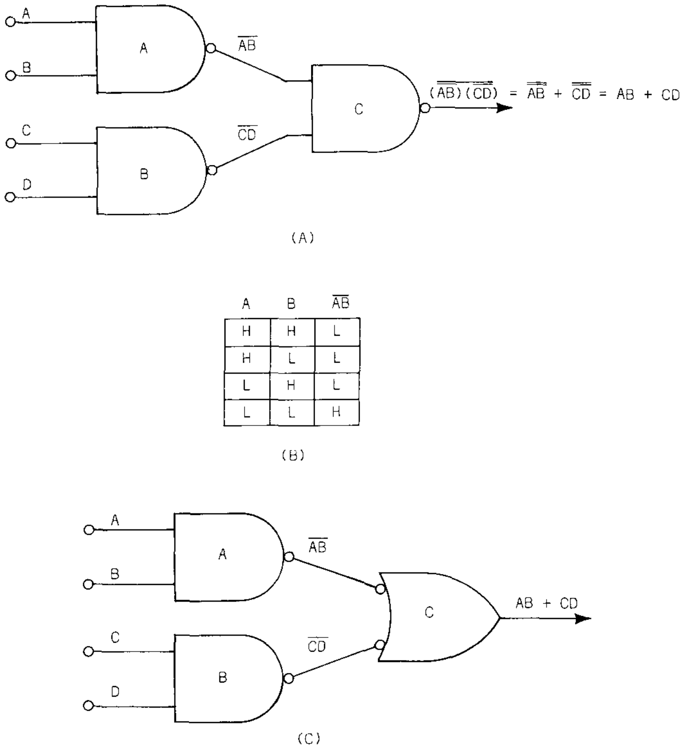


A
B
$\overline{AB}$
$\overline{\overline{AB}} = AB$
A
$\overline{A}$
B
$\overline{B}$
$\overline{\overline{A}\ \overline{B}} = \overline{\overline{A}} + \overline{\overline{B}} = A + B$

(A) AND

(B) OR

Fig. 4-7. Implementing with NAND gate.


A
B
C
D
A
B
$\overline{AB}$
$\overline{CD}$
C
$\overline{(\overline{AB})(\overline{CD})} = \overline{\overline{AB}} + \overline{\overline{CD}} = AB + CD$
(A)
(B)
A
B
C
D
A
B
$\overline{AB}$
$\overline{CD}$
C
AB + CD
(C)

A	B	$\overline{AB}$
H	H	L
H	L	L
L	H	L
L	L	H

Fig. 4-8. Complex NAND circuits.

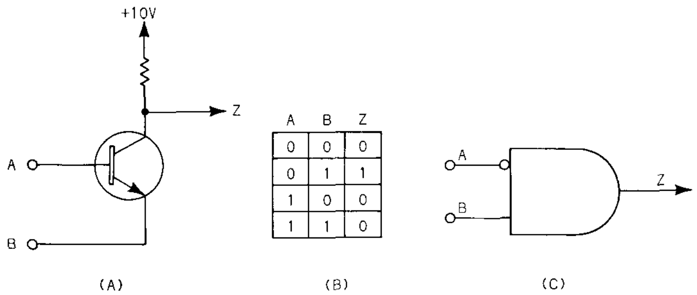
NAND as inverted input OR

Sometimes a NAND gate is shown as an inverted input OR gate. See Fig. 4-8A. The truth table for a NAND gate indicates that with A and B both true a false output results. The same applies to gate B. Here we have a situation in which true inputs give a false output. Gate C responds to the false output levels of gates A and B. Re-examine the truth table for a NAND gate replacing true and false with the negative logic voltage levels (Fig. 4-8B), we find when any NAND gate input is high the output is low. Only if both inputs are low is the output high. Using positive logic instead of negative logic we could say that this gate acts as a positive logic NOR gate. In certain schematics, as in Fig. 4-8C, the symbol for Gate C is replaced by an inverted input OR symbol. A NAND B and C NAND D are inverted before being OR'd. The final result is A AND B OR C AND D. Fig. 4-8A also gives this result if we apply DeMorgan's theorem to the output. The difference is only in the symbology used. The actual logic operation is the same.

The configuration in Fig. 4-8 appears several times in the Tektronix Type 230 and Type 240 instruments. When using NAND gates or NOR gates a system of mixed logic actually occurs within the diagram. The true inputs for a NAND gate give a false output. This operation can be called an inversion in logic values. Thus, many observers say that after passing through a NAND gate (if the input is in a negative logic assignment) the output has a positive logic assignment. In the case of Fig. 4-8, the system inputs are in a negative logic environment. They are inverted once by the first two NAND gates and inverted again by (negative logic NAND or positive logic NOR) gate C. The output level emerges in a negative logic environment. This is the case for many Tektronix digital instrument circuits. The actual logic assignments are fixed in Tektronix diagrams.

inhibitor

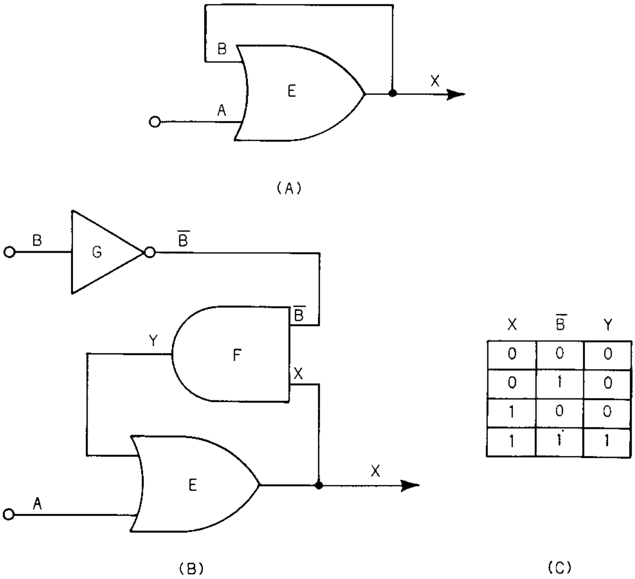
Fig. 4-9A illustrates a transistor whose input terminals are the emitter and the base. In this case, input A must be high and input B must be low for the transistor to be on. The inputs required have opposite levels. The truth table for such a gate in a negative logic environment is shown in Fig. 4-9B. The output is zero (or false) except when A is zero and B is one. The logic symbol for such a gate is shown in Fig. 4-9C. The equation for such a gate is $Z = \bar{A}B$. Occasionally we use such a gate where it is desired to hold one input false and disable the entire gate. Here, if input B is held false, the output at A has no control over the gate. In this sense, then, input B is termed an inhibiting input and the gate is often called an "inhibitor." At Tektronix we consider this type of gate a special form of AND or NAND gate with mixed-logic inputs. Examples of this gate appear in both Type 230 and Type 240 instruments.



A	B	Z
0	0	0
0	1	1
1	0	0
1	1	0

Fig. 4-9. An inhibited gate.

The logic operations discussed to this point are essentially single action in nature. A set of logic signals is applied to a set of logic decision devices which proceed to generate a single result. Many digital systems require nothing more. Others, perhaps the majority, require a series of such operations in sequence. To provide a sequential action, a device is required which has a memory. One which will remember the results of a logic operation for later use. A counting circuit is an example. When counting from one to ten the counter must remember how many units have already been counted. At the fourth count, to realize that this *is* the fourth count, the counting circuits must remember that three prior counts have been made.



(A)

X	$\overline{B}$	Y
0	0	0
0	1	0
1	0	0
1	1	1

(B)

(C)

Fig. 4-10. Evolving a flipflop.

A simple way of obtaining a memory device uses an OR gate. See Fig. 4-10A. OR gate E has a connection from its output back to one of the inputs. Assume that X initially equals zero. If X is zero then A must be zero. If A is now made equal to one, then X must equal one. If X equals one, input B receives the one and regardless of whether A returns to zero the output remains at one. The circuit as shown is impractical because there is no way of forcing X back to the zero state. The gate can be forced back to zero if some means is provided for breaking the feedback path. See Fig. 4-10B. Here the feedback path contains AND gate F. The AND gate F has inputs $\overline{B}$ and X and output Y. A truth table for gate F is shown in Fig. 4-10C. Examining the table Y equals X if $\overline{B}$ is one. If $\overline{B}$ is zero, however, the output is zero. By making $\overline{B}$ equal to zero we force the OR gate E back to zero if A is zero at that time. The logic equation for this circuit would be $Y = \overline{B} \cdot X$. Looking at the overall circuit a one input at A forces the output X to one. A one at B, inverted by gate G, forces the output to zero.

The above is an electronic equivalent to a toggle switch. Flip the switch one way to on, flip the switch the other way to off. Similarly make B equal to one and X flips to zero; make A equal to one and X flips to one. This leads to the definition for a binary memory unit. Such a binary unit has two control inputs. A true level at one input forces the output of the device to one. The other input having a true input forces the output to the opposite state.

The circuit of Fig. 4-10B is bistable. If input A goes to one, X goes to one. If input B goes to one, X goes to zero. This action is similar to an FF (flipflop). An FF can be implemented by using vacuum tubes, transistors, tunnel diodes, magnetic cores or any other two-state active device.

flipflop

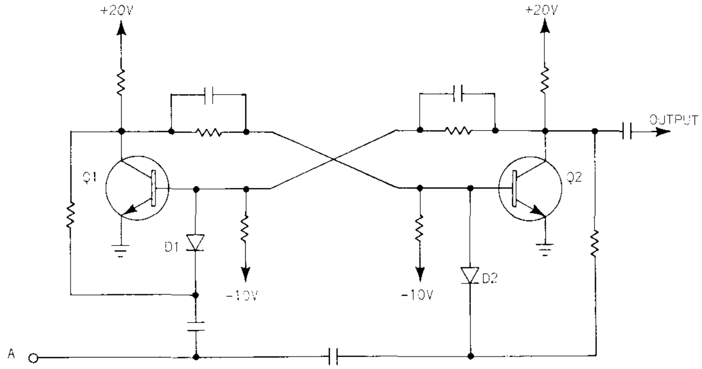


Fig. 4-11. T flipflop.

toggle flipflop (T FF)

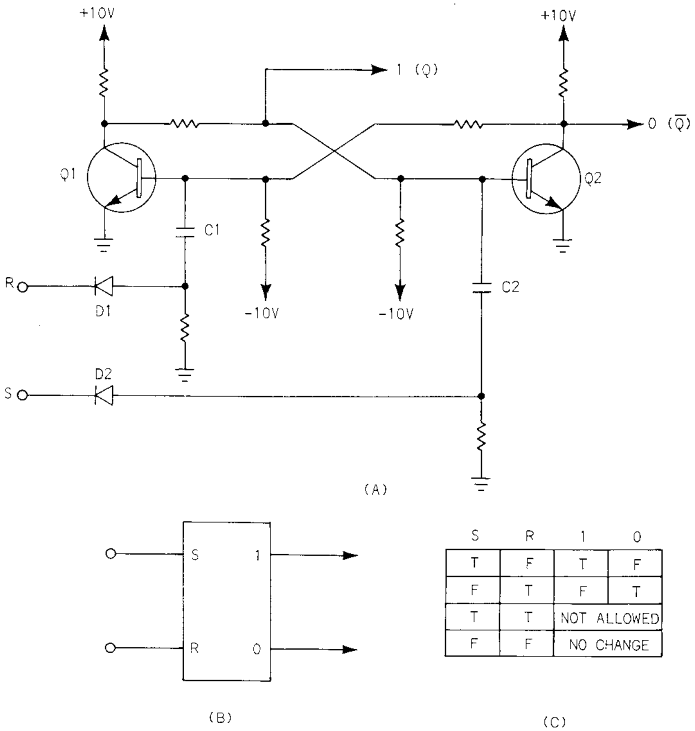
A commonly encountered type of FF using transistors is shown in Fig. 4-11. A trigger pulse introduced at A is differentiated and then conducted via diode D1 and D2 to Q1 and Q2 bases. D1 and D2 polarize the pulse so that only the negative edge of the pulse appears at the bases. The NPN transistors are turned off by a negative edge. Any time a trigger pulse is coupled into input A the flipflop will change state. Since this action is analogous to toggle-switch action, the circuit is called a toggle flipflop (T FF). The T FF has a major shortcoming. The state of the FF after a trigger is applied cannot be accurately known unless the present state is known.

set-reset flipflop (RS FF)

Another common FF is the "Set-Reset" (RS FF). A representative transistor RS FF is shown in Fig. 4-12A. This is similar to the T FF except that there are two input terminals. This circuit is predictable for three of four input conditions.

The inputs labeled R and S are called the Reset and Set inputs, respectively. Two rules for RS flipflops are "Set to one" and Reset to zero." Set to one means an input signal (negative step here) to the Set terminal switches the circuit to a known condition called the *One state*. Reset to zero means that an input signal (negative step) to the Reset input switches the flipflop to the opposite condition called the *Zero state*. It remains to define the output states of the circuit. As an example, consider Fig. 4-12A. A negative edge to the S input couples through D2 and C2 to turn Q2 off. The collector of Q2 goes false and turns on Q1. The collector of Q1 goes true. This output is therefore the one output. Thus, a Set input signal places the flipflop in the one state. A negative edge to the R input switches the 1 output to zero.

The collector of Q2 always logically complements the 1 output and is called the 0 output. In Fig. 4-12B we show the logic diagram symbol for an RS flipflop. A box symbol shows two inputs on the left, two outputs on the right. Tektronix diagrams always show the Set input opposite the 1 output. The Reset appears opposite the 0 output. The rule that an input to R causes the 1 output to go to 0 could also be stated R causes the 0 output to go to 1. The RS flipflop truth table (Fig. 4-12C) shows that all input conditions are covered except when S and R inputs receive simultaneous negative edges. The next state of the FF cannot be predicted and is ambiguous. Since this is true, for the RS flipflop simultaneous R and S inputs are commonly called "not allowed" or "forbidden" combinations. The RS flipflop is used in logic situations which do not include the possiblility of simultaneous Set and Reset inputs.



(A)

(B)

S	R	1	0
T	F	T	F
F	T	F	T
T	T	NOT ALLOWED	
F	F	NO CHANGE	

(C)

Fig. 4-12 The RS flipflop

The circuit of Fig. 4-13A combines the features of the toggle and RS FF's. This particular circuit using PNP transistors responds to positive signals at all inputs. The Clock input responds only to positive-edge signals.

A logic symbol (drawn for negative logic) appears in Fig. 4-13B. The small circles at S and R inputs indicate logic inversion at that point. This should be interpreted to mean that a false level is inverted to become a true level within the box. The 1 and 0 outputs also have circles. From a negative-logic viewpoint these are $\overline{1}$ and $\overline{0}$ outputs. A $\overline{\text{Set}}$ signal (the bar indicates that the set input is the high level) switches the FF to a $\overline{1}$ condition. See the truth table of Fig. 4-13C.

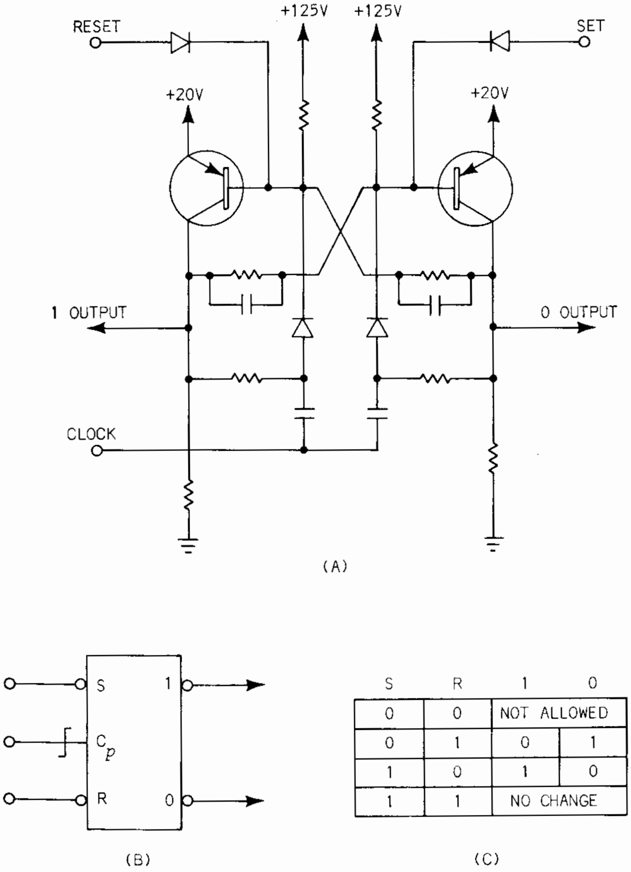
The symbol at the C_p (Clock pulse) input indicates that this input responds to a positive edge signal only.

clocked
RS FF

The clocked RS FF is used in counting circuits. In such circuits the FF operates in a clocked mode for a period of time. Afterwards the FF must be returned to a known condition using either R or S input.

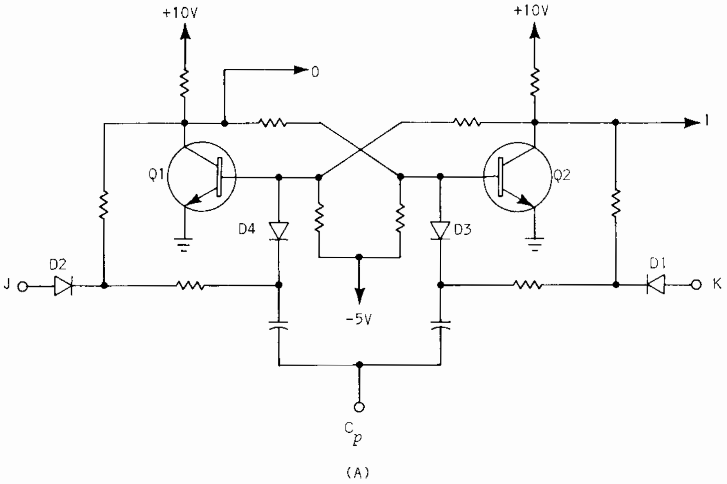
clocked
JK FF

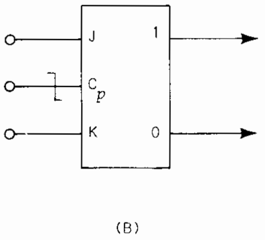
The JK FF has no ambiguous states. When a one is applied to the J, the flipflop is switched to the one state. With a one at K the flipflop is switched to the zero state. If ones are applied to both J and K the FF switches to its complement state. Many JK flipflops are supplied with two or more J inputs and two or more K inputs. Frequently one J and one K input are connected together and called the clock input. This input is usually labeled C_p on Tektronix logic diagrams.



S	R	1	0
0	0	NOT ALLOWED	
0	1	0	1
1	0	1	0
1	1	NO CHANGE	

Fig. 4-13. The clocked RS FF.





J	K	C_p	1	0
F	F	F	NO CHANGE	
F	F	T	NO CHANGE	
F	T	F	NO CHANGE	
F	T	T	1	0
T	F	F	NO CHANGE	
T	F	T	0	1
T	T	F	NO CHANGE	
T	T	T	COMPLEMENT	

(C)

Fig. 4-14. Clocked JK FF.

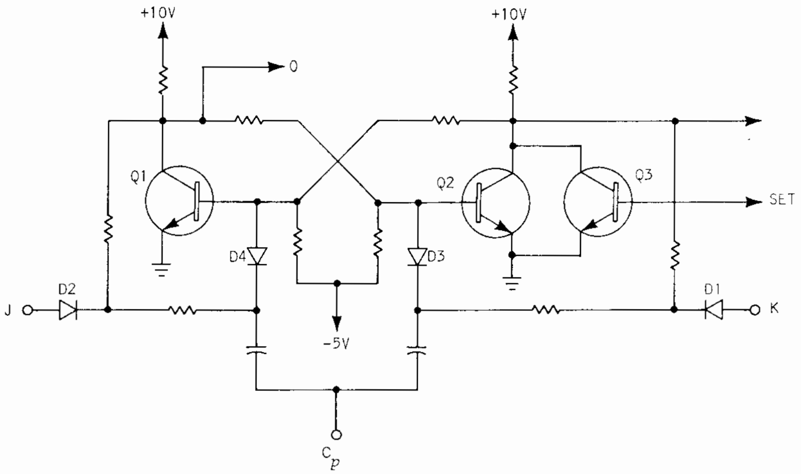


Fig. 4-15. Clocked JK FF with Set input.

Fig. 4-14A shows a JK flipflop using discrete components. The circuit is similar to that for the T FF. Diodes D1 and D2 provide a way of inhibiting C_p input signals. For example a positive level applied to the J input causes D2 to conduct. The positive level coupled through D2 holds D4 off. D4, when off, blocks a negative edge signal to Q1 base. Q1 cannot be turned off. The C_p signal can pass through D3 turning Q2 off (if not already off).

A positive voltage at K couples through D1 to D3. D3 inhibits a C_p signal to Q2. This inhibiting action can be viewed as a way of steering the flipflop to a desired condition. If both J and K are held false the flipflop will hold its present condition when a clock input occurs.

One mode of operation puts J and K at one. For the negative logic assignment used at Tektronix the inputs are at 0 V. J and K are electrically disconnected from D3 and D4, the trigger gates. The operation is now the same as for the T FF. The state of the flipflop changes for each negative edge into the C_p input. For this circuit, just pulling J false or K false will not change the state of the flipflop. The false J or K inputs have no effect until a clock is applied to C_p. This is typical of all clocked JK flipflops. The reader may determine the operation of the flipflop by referring to the truth table of Fig. 4-14C. The logic symbol for a JK flipflop appears in Fig. 4-14B. The JK flipflop has no ambiguous states. The designer can always control the output state of the flipflop regardless of the combination of input signals. For this reason the JK flipflop is used more frequently than any other type.

In certain circuits the designer requires an additional input which overrides the J, K, and clock inputs. The circuit of Fig. 4-14A may have an extra transistor connected as shown in parallel with Q2. See Fig. 4-15. The base of Q3 would be called a set input. By putting a false level on Q3 the 1 output goes true regardless of any other inputs to the circuit.

set input

5

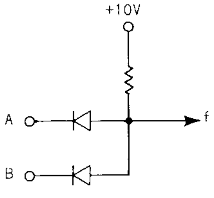
IMPLEMENTING LOGIC FUNCTIONS

Logic functions can be implemented by the use of any component which can act as a switch. This includes mechanical switches, relays, diodes, transistors, vacuum tubes, field-effect transistors; indeed any active device which can turn on and off. The various methods of using the switching devices to implement logic functions are given special names. The name is usually a series of letters taken from the first letter of the first words which name each type of logic circuit. Thus we have such names as DL which stands for Diode Logic, RTL standing for Resistor-Transistor Logic, DTL for Diode-Transistor Logic, TTL for logic systems using Transistor-coupled to Transistor Logic circuits and CML for Current-Mode Logic. Within the digital circuit industry, other initials are commonly used. These, however, are primarily utilized for commercial purposes and are normally found to be modifications of the previously named types of logic systems.

Knowledge of the exact type of logic circuit used is often of minor importance to the user of the completed device. However, this knowledge is very important to the device designer because the families of logic circuits possess various advantages and disadvantages which recommend one over the other. Also, to troubleshoot or circuit trace an existing instrument, a knowledge of the shortcomings of the types of logic circuits becomes important.

diode logic

The semiconductor diode is a two-terminal, nonlinear switching device. It is binary in nature because when forward biased it has low forward resistance, and when reverse biased, it has high reverse resistance. Semiconductor diodes used to perform logic functions were among the first devices utilized in digital circuits, principally because they are small, inexpensive, fast switching and operate at low power levels. The diode, however, is a nonamplifying device, and circuits which employ diodes are usually limited to single logic functions.

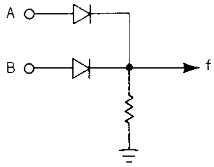


A	B	f
H	H	H
H	L	L
L	H	L
L	L	L

FOR POSITIVE LOGIC f = AB

FOR NEGATIVE LOGIC f = A + B

(A)



A	B	f
H	H	H
H	L	H
L	H	H
L	L	L

FOR POSITIVE LOGIC f = A + B

FOR NEGATIVE LOGIC f = AB

(B)

Fig. 5-1. Diode logic gates.

Practical circuits utilizing diode logic circuits often include active transistors or other amplifying devices to offset diode losses. Two basic diode logic circuits are shown in Fig. 5-1. Fig. 5-1A shows a diode circuit and a voltage-level truth table. This circuit operates as an AND gate in a positive-logic system, and an OR gate for a negative-logic system. Fig. 5-1B, on the other hand, is a positive-logic OR gate and a negative logic AND gate. Since these are universal circuits it is important to realize that the same circuit in one instrument could be an AND gate and could equally well be an OR gate in a different instrument.

diode losses

In either of the circuits of Fig. 5-1, assuming that silicon semiconductor diodes are utilized, approximately 0.6 volts is lost between the input and the output signal levels. If an attempt is made to cascade several diode-logic circuits, it is usually found that enough voltage is lost to render the circuit almost unusable. In early instruments which used this type of logic circuit, an amplifying transistor or vacuum tube was often inserted to replenish circuit losses.

When operated in high-speed logic systems, the diode logic circuit has several shortcomings. One problem is the fact that when a diode is forward biased and an attempt is made to turn it off, it takes an interval of time before the stored charge is fully fan out

swept out of the diode junction. This limits switching speed for the diode. In addition, the diode logic circuit suffers from poor "fan-out" capabilities. Fan out is a term which refers to the number of other logic-device input circuits that a particular logic output circuit may be capable of driving.

The diode logic circuit has poor fan-out capabilities because the diode has low forward resistance and provides little isolation between input and output. The source impedance of the circuit can cause excessive voltage losses. If high-speed logic functions are to be performed, the ability of a diode circuit to charge a capacitive input is limited by the finite forward resistance.

The biggest advantages of diode logic circuits are small size and low cost. Because of these advantages, which often outweigh the disadvantages, frequent use is made of diode logic circuits.

resistor-transistor logic

cutoff

saturation

Resistor-transistor logic circuits use the transistor as an active element. The transistor has several advantages. It provides both voltage and current gain which gives it excellent noise immunity and excellent fan out. The transistor is normally operated as a switching device; that is, it operates between a cutoff condition and a saturated condition. When cut off the transistor has high reverse resistance and when saturated has low forward resistance. Thus it is a good binary element.

time delay

On the other hand, transistors operated into saturation may suffer from several forms of time delay. When turned on, the transistor takes an interval of time before the carrier condition is fully established. A saturated transistor requires a certain amount of turn-off time because of the minority-carrier storage time of the forward-biased collector-to-base junction. This limits the usefulness of the RTL circuit in high-speed logic applications.

common-emitter mode

RTL circuits use transistors in the common-emitter amplifier mode. Because of the bipolar nature of transistors, circuits utilize both NPN and PNP devices.

A representative RTL circuit using a PNP transistor is shown in Fig. 5-2. This is a three-input NAND gate. The three inputs, A, B and C, are each coupled to the base of Q1 by a 12-k resistor. The logic levels for this particular circuit are 0 V = 1, 12 V = 0. Unless A and B and C are all true, the base of Q1 is not negative with respect to its emitter and Q1 remains off. Only with A and B and C at zero volts, can Q1 be on, and the output be false. This is characteristic of a NAND gate.

The major point of interest in this circuit is the voltage level at node M in Fig. 5-2A. To confirm the truth of the previous statements, let us examine the possible conditions for the gate. First we consider the C input true, the other two inputs false. This is the voltage condition shown in Fig. 5-2B. We apply Thevenin's theorem to the two-resistor branch consisting of R1 and R2 which gives the result shown in Fig. 5-2C. Again applying Thevenin's theorem gives the circuit of Fig. 5-2D. R3 in series with the equivalent 4.3 kΩ puts node M at +20.7 volts. This assures that Q1 is off.

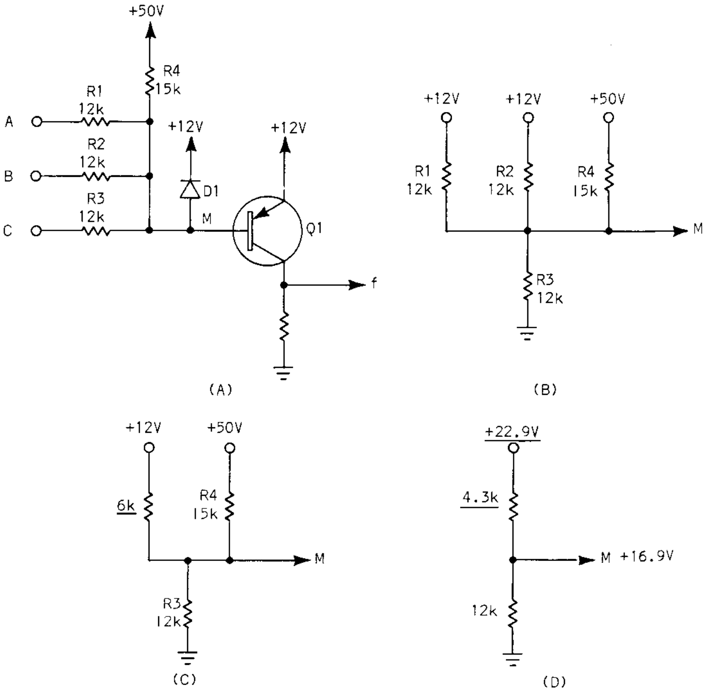


Fig. 5-2. An RTL circuit using a PNP transistor.

If Q1 were to be a silicon planar PNP transistor, it would be necessary to add a clamp-diode D1 at node M to prevent Q1 from going into emitter-base breakdown. This diode would clamp node M at approximately +12.7 volts. Without the diode, node M would be at +16.9 volts.

If we consider two inputs (B and C) true we have the circuit of Fig. 5-3A. This Thevenizes to the simplified circuit in Fig. 5-3B. The resistor ratio places node M at approximately +13.8 volts. This level assures that Q1 is off.

Finally, considering the case where all three inputs are true, we have the circuit of Fig. 5-3C. This simplifies to the circuit of Fig. 5-3D placing node M at approximately +10.3 volts. Q1 is turned on with +10.3 volts at the base.

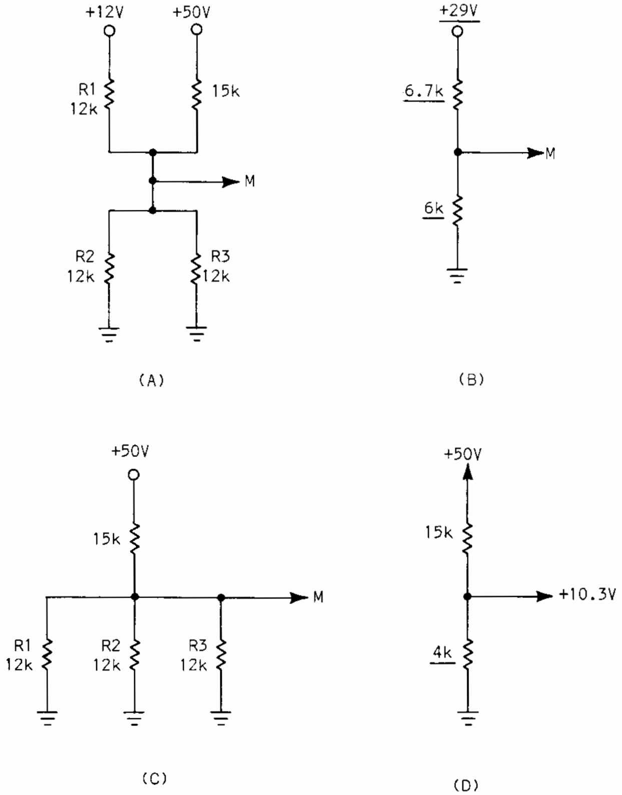


Fig. 5-3.

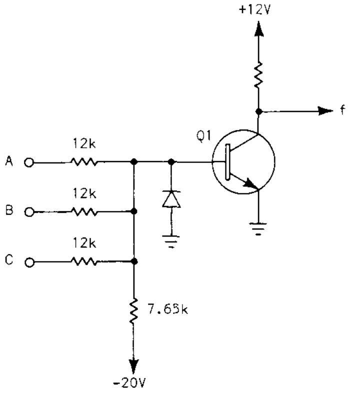
If Q1 were to draw excessive emitter-base current a resistor could be added between node M and the base of the transistor to limit that current. Fig. 5-4 illustrates a similar circuit which performs as a negative-logic NOR gate. Here an NPN transistor is used. Q1 is cut off unless all inputs are false. Study of the voltage-level truth table shows that this is a negative-logic NOR gate. The reader may solve the resistive divider for all possible input combinations using the methods for Fig. 5-3 to convince himself that the circuit operates as stated.

majority logic

Fig. 5-5A shows the symbol of a three-input circuit which performs a majority logic operation. Unlike ordinary Boolean algebra circuits, a majority circuit responds to the majority of its inputs. A truth table for such a function of a three-input majority logic gate is shown in Fig. 5-5B. The output of the gate is 0 unless two or more of the inputs (a majority) are 1, in which case, the output is 1.

minority logic

One of the simplest methods of implementing this function uses RTL with an inverting transistor. The majority output is thus inverted. See Fig. 5-5C. A majority inverted becomes a minority. The output (carry) agrees with the minority of A, B, and C input levels. This composite gate is represented by the symbol of 5-5D. The truth table for the minority gate is shown in Fig. 5-5E.



A	B	C	f
H	H	H	L
H	H	L	H
H	L	H	H
H	L	L	H
L	H	H	H
L	H	L	H
L	L	H	H
L	L	L	H

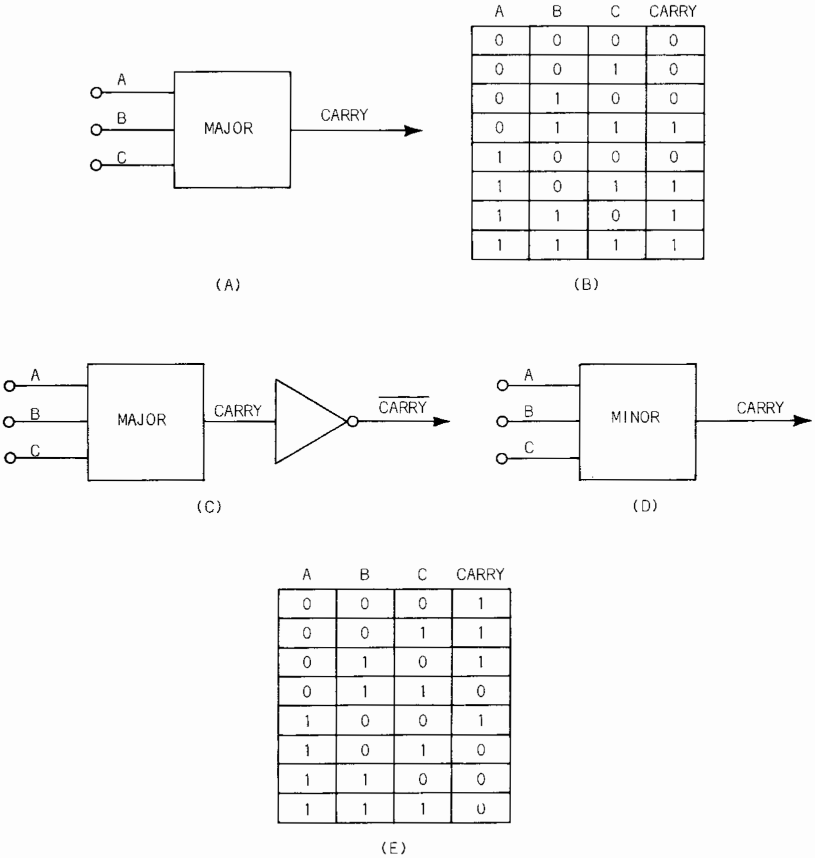
FOR NEGATIVE LOGIC

$f = \overline{A + B + C}$

FOR POSITIVE LOGIC

$f = \overline{ABC}$

Fig. 5-4. An RTL circuit using an NPN transistor.



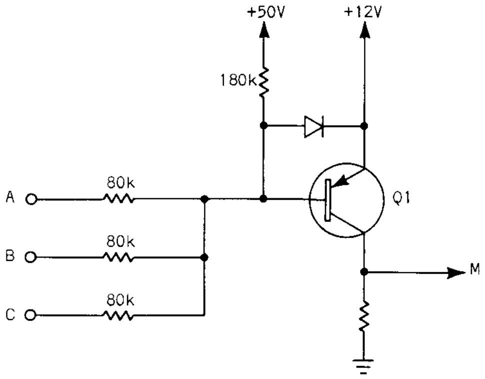
(B)

A	B	C	CARRY
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

(E)

A	B	C	CARRY
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

Fig. 5-5. Majority/minority functions.



(A)

A	B	C	M
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

(B)

Fig. 5-6. A minority circuit.

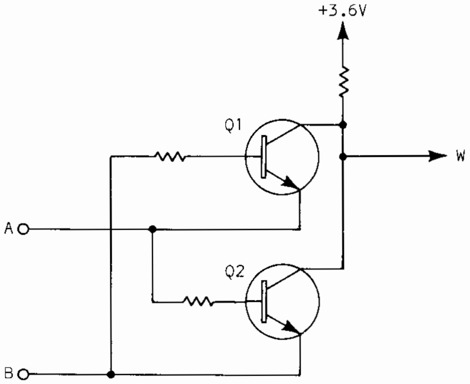
A circuit which performs a minority function appears in Fig. 5-6A. The circuitry appears to be the same as that of a three-input NAND gate. The resistive components are chosen so that Q1 remains off unless at least *two* of the inputs are true. A truth table for the gate is shown in Fig. 5-6B. The gate uses logic levels of 0 = +12 V and 1 = 0 V. The reader may solve the circuit and prove the truth table.

Exclusive-OR gate

The Exclusive-OR gate (also called an anticoincidence gate) delivers a true output if the input states are not identical. The output of the gate is false if the inputs are identical. A typical discrete-component transistor Exclusive-OR gate is shown in Fig. 5-7A. Transistors Q1 and Q2 are connected together in the form of a RTL logic circuit. Input A is connected to the emitter of Q1 and the base of Q2. Input B is connected to the base of Q1 and the emitter of Q2. If inputs A and B are at the same level, both transistors have zero-biased emitter-base junctions and neither transistor is on. Only if the inputs are at opposite logic levels can one or the other of the transistors be on. A truth table for the device is shown in Fig. 5-7B.

diode-transistor logic

The major disadvantage of the RTL circuit is slow operation due to saturation. In addition, given a fixed logic assignment and restricted to a single type of transistor (NPN or PNP), there is no way to obtain both NAND and NOR operations. Thus certain logic expressions cannot be implemented efficiently. Both disadvantages are avoided by combining the diode logic circuit with the transistor.



(A)

A	B	W
0	0	0
0	1	1
1	0	1
1	1	0

(B)

Fig. 5-7. Exclusive OR gate.

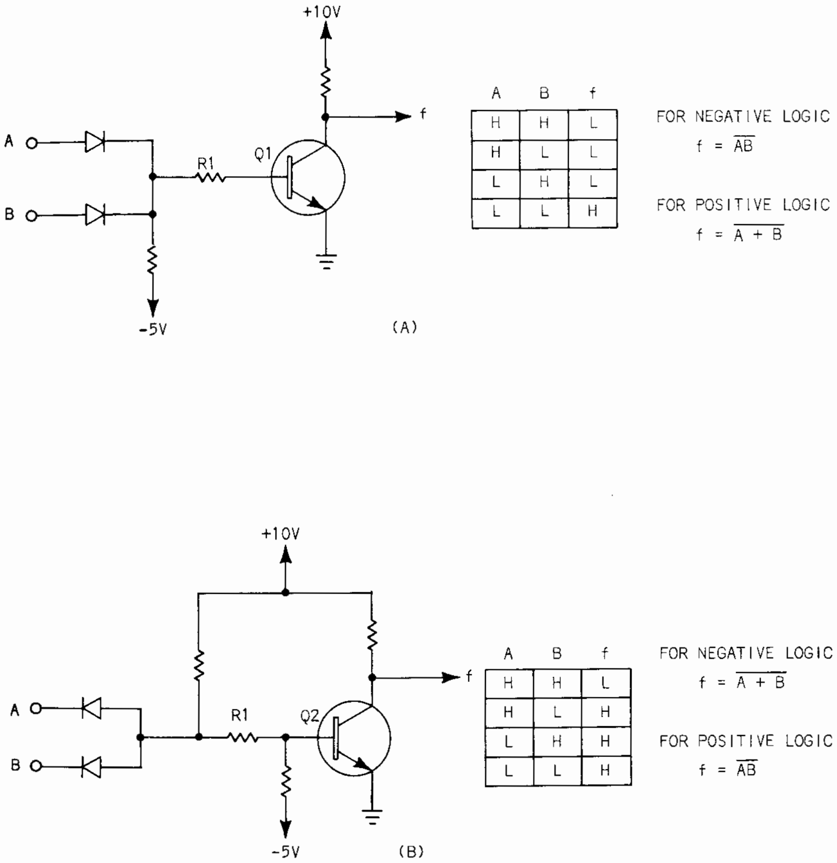
A typical circuit of DTL is shown in Fig. 5-8A. In this example the diodes perform the logic function while the transistor amplifies and inverts the results. Logical 1 equals 0 V and logical zero equals 10 V. The circuit of Fig. 5-8A must have both inputs true for Q1 to be turned off. For all other conditions Q1 base is positive, turning Q1 on. The voltage-level truth table demonstrates that this is a negative-logic NAND gate. Study of the circuit of Fig. 5-8B shows that it functions as a negative-logic NOR gate.

In both figures, an NPN transistor is used. The only difference is the polarity of the series diodes and the pull-up or pull-down voltage required. By proper design of the circuit, particularly the selection of R1, the amount of saturation can be limited. This makes a faster switching circuit. Since diodes are often less expensive than transistors this circuit can be more economical than logic circuits using multiple transistors. The circuits of Fig. 5-8 could also be implemented by PNP transistors.

Direct-Coupled Transistor Logic

Transistor logic circuits of the previous sections utilize the transistor only as an amplifier, the logic function being performed by resistors or diodes. A commonly used family of logic circuits utilizes transistors to perform the logic operation as well as amplify. This family is called Direct-Coupled Transistor Logic (DCTL).

DCTL may be divided into two categories: those using the transistor as an inverting amplifier (common emitter) and those using a noninverting amplifier (emitter follower).



A	B	f
H	H	L
H	L	L
L	H	L
L	L	H

FOR NEGATIVE LOGIC

$f = \overline{AB}$

FOR POSITIVE LOGIC

$f = \overline{A + B}$

A	B	f
H	H	L
H	L	H
L	H	H
L	L	H

FOR NEGATIVE LOGIC

$f = \overline{A + B}$

FOR POSITIVE LOGIC

$f = \overline{AB}$

Fig. 5-8. DTL circuits.

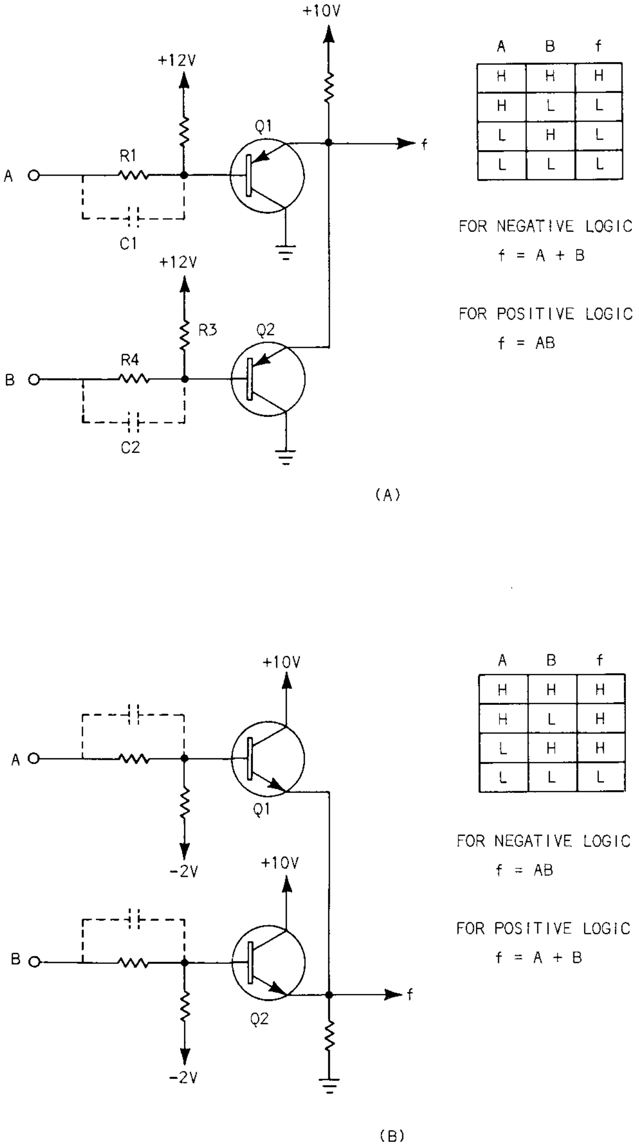
DCTL emitter follower

Fig. 5-9 shows two DCTL circuits utilizing both NPN and PNP transistors as emitter followers. Fig. 5-9A shows a two-input negative-logic OR gate (positive-logic AND gate). Here a true input (0 V) at input A or input B biases either Q1 or Q2 on, placing the output at approximately 0 V, logical one. Therefore, if A is true or B is true (or both) the output is true. Examining the A input, resistor R1 limits the emitter-base current drawn through Q1. R2 returned to +12 V assures turn off of Q1 when the A input is false. Capacitor C1 is often included to help remove the stored charge when turning off the transistor, thus improving the switching speed of Q1. Input B is similar.

Fig. 5-9B shows a two-input logic gate using NPN transistors. With negative-logic assignment it performs as an AND gate. If A is true and B is false the output is false. The high at input B keeps Q2 on. Q2 when turned on pulls the output to +10 volts. Only with both A and B true (Q1 and Q2 both turned off) can the output be true. The gate performs an AND function.

The emitter follower has a very low output impedance. Therefore, this type of circuit is characterized by having good fan-out capabilities. On the other hand, a serious disadvantage is that an emitter follower has no voltage gain. In fact, there is a voltage loss, depending upon the type of transistor and the values of the resistances which are used. This means that only a limited number of stages can be cascaded. At intervals in a cascade of circuits an inverter or restoring circuit must be included to reestablish the proper signal level.

Another disadvantage of the emitter-follower logic gate is that a particular type of transistor emitter-follower will charge faster in one direction than the other. A PNP-type circuit such as in Fig. 5-9A will have a faster falltime than risetime. The reverse is true for the NPN circuit of Fig. 5-9B. This can be offset at the expense of additional transistors and components by using a complementary emitter-follower



A	B	f
H	H	H
H	L	L
L	H	L
L	L	L

FOR NEGATIVE LOGIC

$f = A + B$

FOR POSITIVE LOGIC

$f = AB$

(A)

A	B	f
H	H	H
H	L	H
L	H	H
L	L	L

FOR NEGATIVE LOGIC

$f = AB$

FOR POSITIVE LOGIC

$f = A + B$

(B)

Fig. 5-9. DCTL noninverting circuits.

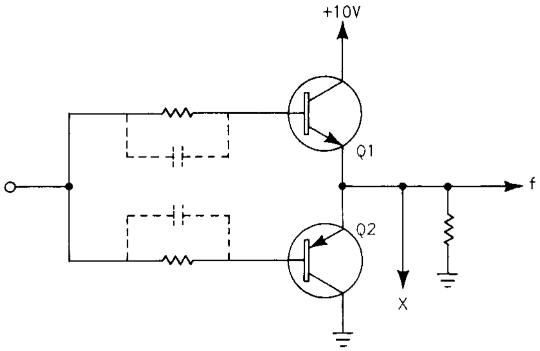


Fig. 5-10. Complementary emitter follower.

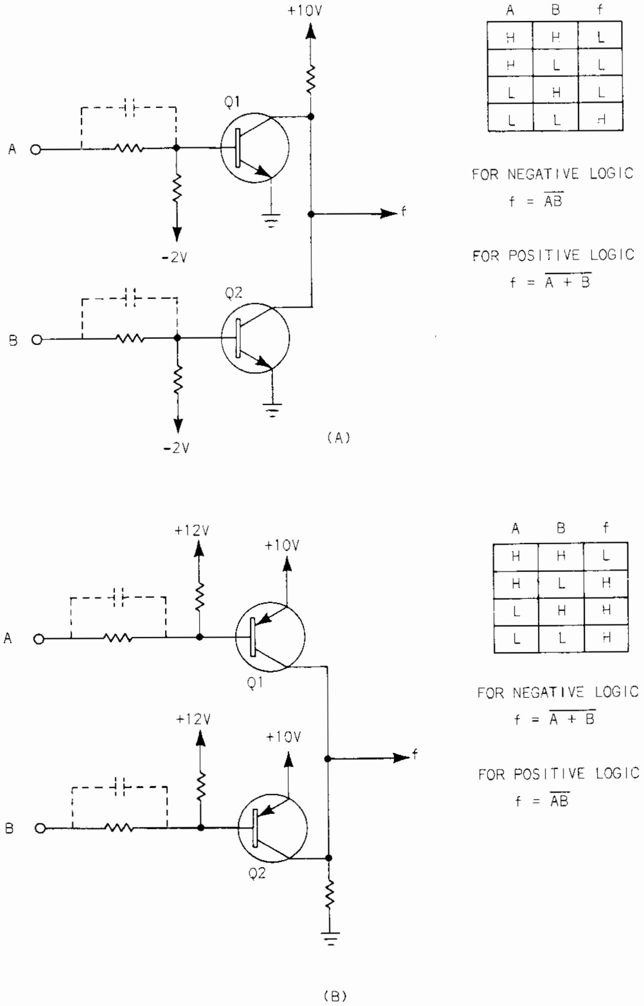
arrangement. Fig. 5-10 shows part of an emitter-follower logic circuit which has been complemented. Q1 is an NPN transistor and Q2 is a PNP transistor. With this configuration, no matter whether the output signal is going from true to false or from false to true, either Q1 or Q2 is turning on. This improves the risetime and falltime of the circuit. This circuit would have to be duplicated for each gate input required. Such additional circuits would be connected to the output at point X.

DCTL
common
emitter

Fig. 5-11A shows two NPN transistors which are connected as common-emitter amplifiers sharing a common-collector load resistor. Unless both Q1 and Q2 are turned off, the output will be logically true. If Q1 and Q2 are both off, then the output would be false. This is characteristic of a negative-logic NAND gate. In a positive-logic environment this circuit is a NOR gate. A voltage-level truth table is shown for the gate.

A similar circuit using PNP transistors is shown in Fig. 5-11B. With either input A or input B or both at logical one (0 V), Q1 or Q2 or both are turned on and the output is zero (+10 V). Only if inputs A and B are both false can the output be true. This is characteristic of a negative-logic NOR gate.

Instead of having the transistors connected in parallel they can be connected in series. When two NPN transistors and a collector resistor are connected in series, as shown in Fig. 5-12A, the circuit behavior is similar to that in Fig. 5-11B. This is a negative-logic NOR gate. If A is true or B is true, one or the other of the transistors is turned off and f is false. The output f drops to a logical one level only if both transistors are on. For both transistors to be on both A and B would have to be high or false. At that time the output f is true.



A	B	f
H	H	L
H	L	L
L	H	L
L	L	H

FOR NEGATIVE LOGIC

$f = \overline{AB}$

FOR POSITIVE LOGIC

$f = \overline{A + B}$

A	B	f
H	H	L
H	L	H
L	H	H
L	L	H

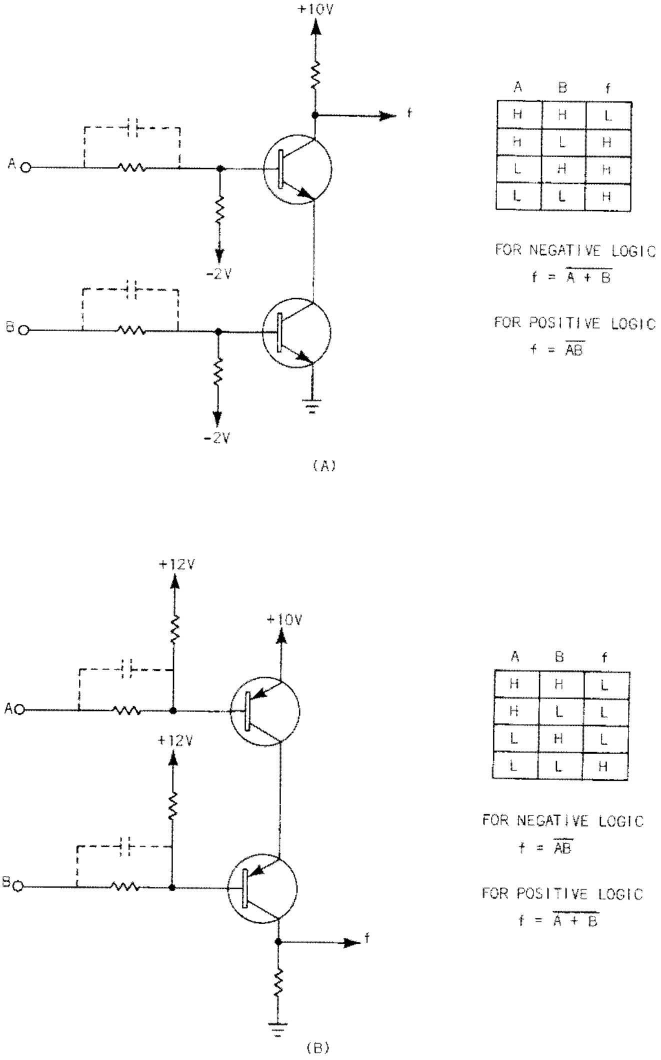
FOR NEGATIVE LOGIC

$f = \overline{A + B}$

FOR POSITIVE LOGIC

$f = \overline{AB}$

Fig. 5-11. Parallel-DCTL inverting circuits.



(A)

A	B	f
H	H	L
H	L	H
L	H	H
L	L	H

FOR NEGATIVE LOGIC
$f = \overline{A + B}$

FOR POSITIVE LOGIC
$f = \overline{AB}$

(B)

A	B	f
H	H	L
H	L	L
L	H	L
L	L	H

FOR NEGATIVE LOGIC
$f = \overline{AB}$

FOR POSITIVE LOGIC
$f = \overline{A + B}$

Fig. 5-12. Series-DCTL inverting circuits.

Connecting PNP transistors and the collector load in series results in a circuit as shown in Fig. 5-12B. If A and B are both low, both transistors are on, and the output is logical zero. For any other combination of inputs, one or the other or both transistors are off and the output is one. For negative logic, this is a NAND gate.

Comparing circuits we see that if NPN transistors are operated in parallel, a negative-logic NAND gate occurs. If they are operated in series, the negative-logic NOR gate occurs. For PNP transistors operated in parallel, a negative-logic NOR gate occurs and for PNP in series a negative-logic NAND gate occurs. Thus, either NAND or NOR functions can be implemented using only NPN or PNP transistors.

It is possible in any of the above circuits to have more than two transistors in parallel or in series, thus, increasing the number of inputs to the gate. Care must be taken in the series form to avoid too many transistors in series since the saturation resistances add.

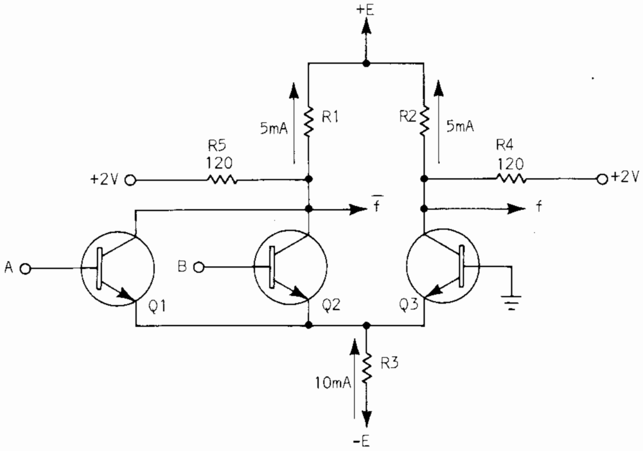
The major limitation of DCTL circuits is the relatively slow turnoff time of a saturated transistor. This has resulted in many ingenious designs which prevent the transistor from saturating; the minority-carrier storage time being thus avoided. To examine these is beyond the scope of this book.[1]

current-mode logic

The principle disadvantage of transistor-coupled logic circuits is that the transistor is operated in a saturated mode. A saturated transistor suffers the problem of storage time as a speed-limiting factor. Another form of transistor logic circuit is "emitter-coupled logic" often called current-mode logic (CML). The family was designed as a nonsaturating form of logic which eliminates transistor storage time as a speed-limiting factor. This permits extremely high-speed operation.

The term "current mode" does not have a well-defined meaning. In general, it refers to circuits with small signal-level changes and where nearly equal currents switch from one path to another. CML circuits combine features of previously described logic families but use emitter coupling between amplifier circuits.

[1] Yaohan Chu, *Digital Computer Design Fundamentals* (New York: McGraw Hill, 1962), p 185



(A)

A	B	f	$\bar{f}$
H	H	H	L
H	L	H	L
L	H	H	L
L	L	L	H

(B)

FOR NEGATIVE LOGIC

$f = AB$

$\bar{f} = \overline{AB}$

FOR POSITIVE LOGIC

$f = A + B$

$\bar{f} = \overline{A + B}$

(C)

Fig. 5-13. A CML circuit.

Fig. 5-13 illustrates the basic current-switching technique as it first appeared. Input signals A and B are applied to the bases of Q1 and Q2. The base of Q3 is connected to ground. Both positive and negative power supplies are large relative to signal potential levels, so nearly constant current may be assumed to flow through R1, R2 and R3. The values of R1, R2 and R3 are chosen so that the current through R3 is about twice that through either R1 or R2. For example, assume the current through R3 to be 10 milliamperes. Inputs A and B couple to the bases of Q1 and Q2. An output, f, is taken from the collector of Q3.

If either of inputs A and B is false (approximately +2.6 V) the emitters of Q1 and Q2 are pulled up to approximately +2 V. The emitter of Q3, also at +2 V, cuts off Q3. With Q3 off, output f rises to approximately +2.6 V. The 5-mA current demand of R2 causes 5-mA current flow through R4. The voltage drop of 0.6 V across R4 places f at +2.6 V.

If inputs A and B are both true, (approximately -0.6 V both Q1 and Q2 cut off. This is because the emitter buss is clamped at approximately -0.6 V by the emitter-base junction of transistor Q3. This leaves transistors Q1 and Q2 with zero bias. In this situation Q3 is conducting the ten milliamperes through R3. We assumed a constant 5 milliamperes through R2, so the additional 5 milliamperes of current flows through the 120-ohm resistor to the +2-V supply. Five milliamperes through the 120 ohms puts the collector of Q3 at approximately +1.4 V. Note that Q3 is not saturated.

Fig. 5-13B shows a voltage-level truth table for this circuit. For negative logic with the output taken at the collector of Q3, the circuit functions as an AND gate. For positive logic it is an OR gate. An additional output can be taken from the collectors of Q1 and Q2. This second output will be inverted from the output taken at the collector Q3. Therefore, it is labeled $\overline{f}$, the complement of f. We find that for negative logic $\overline{f}$ is equal to $\overline{AB}$, and for positive logic the $\overline{f}$ output is equal to $\overline{A + B}$.

The advantage of this type of circuit is that neither transistor is saturated. The disadvantages are the multiple power-supply requirements and the shift in output levels from input to output. One solution to the output-level problem alternates NPN and PNP transistor gates from one stage to another within a switching network. PNP transistors in a circuit of this type would perform OR/NOR logic in a negative-logic environment. The PNP circuits would produce a bias shift in the opposite direction correcting the NPN circuit offset.

This availability of AND as well as OR circuits may simplify logic design to some extent; however, dummy stages, to provide offset correction, are often required when only one or the other function is required.

Adding transistors in parallel with Q1 and Q2 expands the input capabilities of this gate. Whether the output is taken as f or $\bar{f}$, a similar source impedance is available. The 120-ohm resistor in each collector circuit guarantees that the output impedance is about 120 ohms. Therefore this type of circuit has fairly good fan-out capabilities. The fan out is improved by the high gate input impedance seen at A or B. A particularly important point is that the power supplies are loaded approximately the same amount regardless of the gate's activity. This constant loading of the power supplies gives good noise immunity. The major circuit disadvantage is higher cost. The antisaturation of this circuit was particularly important with early transistors where minority-carrier storage time was predominant in limiting speed. Since newer transistor types are faster, this version of current-mode switching is no longer as widely used.

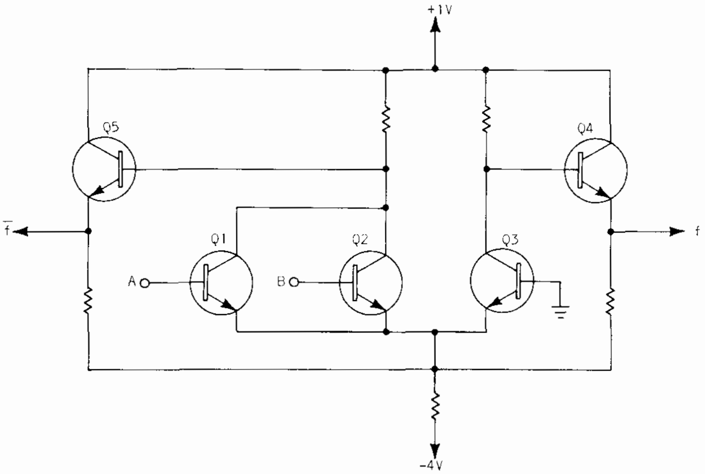
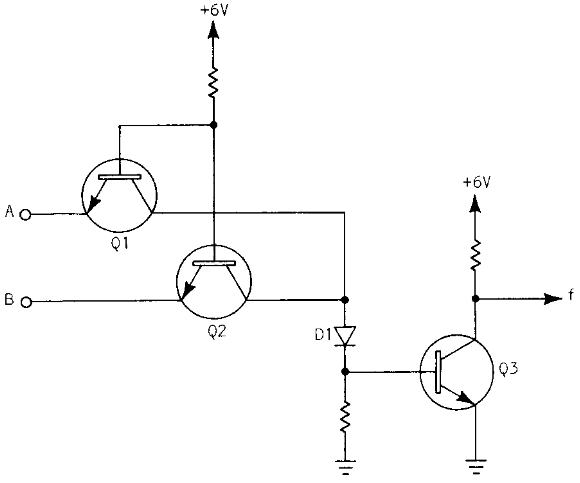


Fig. 5-14. A CML gate.

Another version of CML is illustrated in Fig. 5-14. In this circuit, transistors Q1, Q2 and Q3 function in essentially the same manner as in the previous circuit. However, here no attempt is made to limit the collector swing of the transistor to prevent saturation. The power supply used is relatively small so that the currents are not particularly constant. High speed depends upon the use of fast transistor types. Fan-out capabilities of this circuit are improved principally because of the use of transistors Q4 and Q5 as emitter followers. These give this circuit a significantly lower output impedance. The typical output impedance for an emitter follower is 2 ohms or less.

The circuit maintains the relatively high input impedance at inputs A and B by the common-emitter configuration of the transistors. The collector of Q3 cannot be any more positive than the +1-V supply to which it is connected. If Q3 is saturated, the collector swing is limited in the negative direction to approximately 0.6 V by the collector-to-base junction of Q3. These voltages are offset by the bias of output emitter-follower Q4. In this particular circuit the output levels would be +0.4-V false and -1.2-V true, giving a logic-level swing of about 2 V. At least one major integrated-circuit manufacturer uses the circuit of Fig. 5-14 to form a large part of their logic-circuit product line. See discussion of Motorola integrated circuits in Chapter 6.



(A)

A	B	f
H	H	L
H	L	H
L	H	H
L	L	H

(B)

FOR NEGATIVE LOGIC

$f = \overline{A + B}$

FOR POSITIVE LOGIC

$f = \overline{AB}$

(C)

Fig. 5-15. A TTL gate.

transistor-transistor logic

Fig. 5-15 shows what is commonly called a Transistor-Transistor Logic circuit or TTL. The circuit may be thought of as being derived from DTL. Here Q1 and Q2 are connected as common-base amplifiers. The inputs are made to the emitters of both transistors. The collectors of Q1 and Q2 are paralleled and connected to the base of Q3, the output transistor, which operates as an inverting amplifier. This circuit may be thought of as being derived from the DTL circuit of Fig. 5-8.

A disadvantage of the DTL circuit is that when the transistor is changed from conducting to cutoff condition, the diodes connected to the base present a high resistance to the flow of current from the base. Consequently, an additional resistor is needed to drain away the charge. In the circuit of Fig. 5-15, the transistors being operated as common-base amplifiers may be considered as quasi-diodes. When input signals are positive, no current can flow to the bases of Q1 and Q2. The polarity of the voltage across the collector-to-base junctions of these transistors is then opposite the normal polarity for transistor operation. The transistors operate as a low-resistance path for the flow of current from the base of Q3 to the power supply. If input signal A or B is zero volts, either Q1 or Q2 will be on and saturated. Assume zero volts at input A. This places the base of Q1 at approximately +0.6 V. The collectors of Q1 and Q2, tied together, pull to the same voltage (+0.6 V).

Diode D1 will be forward biased (assuming that all transistors and the diode are silicon) and the base of Q3 is at approximately 0 V. Q3 is cut off. For the circuit, if either A or B is true, the output transistor is cut off.

A voltage-level truth table is shown in Fig. 5-15B. By utilizing high-frequency transistors, storage problems are reduced in the gate. TTL circuits are most frequently encountered in IC form.

6

IMPLEMENTING LOGIC CIRCUITS USING INTEGRATED CIRCUITS

IC fabrication

The digital integrated circuit (IC) or chip is a device which contains complete digital circuits. The entire circuit, including diodes, resistors, capacitors and transistors, is made as an entity, in much the same way as a single transistor is made. The present trend indicates that in the near future, digital instruments will utilize virtually 100% integrated-circuit logic. This is because of the relative reliability of IC logic devices; their small size, the lower cost (since assembly labor is almost entirely automated), and universal availability.

Examination of the different integrated-circuit logic devices available on the market today shows that each integrated-circuit is based upon one or more of the previously mentioned families of logic circuits. It has been found that certain of these families, such as the DTL, are much easier to fabricate using integrated-circuit techniques. Other families, such as RTL, do not lend themselves to IC techniques. Therefore, many of the earlier types of logic families are fast disappearing.

The first successful IC used DTL. However, it is no more expensive in the final design of an IC to put a transistor on the chip than a diode. By using logic families with more transistors, the circuit losses and slow speed of diodes is eliminated. The latest IC's take full advantage of the increased speed, reliability, and fanout of the transistor.

Tektronix digital instruments at the present time utilize IC's manufactured by Fairchild and Motorola. We shall describe several of the types of IC's supplied by each manufacturer.

Fairchild 914

The Fairchild IC's presently used by Tektronix are primarily of the RTL family. An example is the Fairchild Type µL 914 dual two-input NAND/NOR gate. The schematic is shown in Fig. 6-1A. Pin numbers are indicated. There are two independent two-input gates on the chip. The chip is available in an 8-lead TO-5 size epoxy package. The device is compatible with other Fairchild 900-Series IC's.

Each input is resistor-coupled to the base of a transistor. The collectors of the transistors are parallel connected. This type of logic circuit was previously referred to as DCTL. However, throughout the IC fabrication industry this family of logic is now called RTL. RTL was one of the first practical digital-logic families to be produced in IC form. It does suffer the limitations of switching transistors because the transistors operate in saturated mode. Typical delay time is 12 ns. The V_{CC} is +3.8 V for all Fairchild 900-Series logic chips.

914 as NOR gate

The logic symbol of the 914 is shown in Fig. 6-1B. For positive logic each gate acts as a NOR gate; for negative logic each gate acts as a NAND gate. If pins 6 and 7 are tied together a single four-input NAND gate results.

phantom OR

Fig. 6-2 illustrates the two sections of a 914 with the outputs in parallel. The symbol for a phantom OR (sometimes called a wired OR) appears at the junction between pins 6 and 7. Interpret the phantom OR by saying, "If pin 6 is true or pin 7 is true, the output is true."

914 as NAND gate

The logic diagram may show a symbol for the 914 as in Fig. 6-2A or in 6-2B; a single NAND gate with four independent inputs. The presence of two connections at the output may be inferred by pin numbers 6 and 7 shown at the output connection.

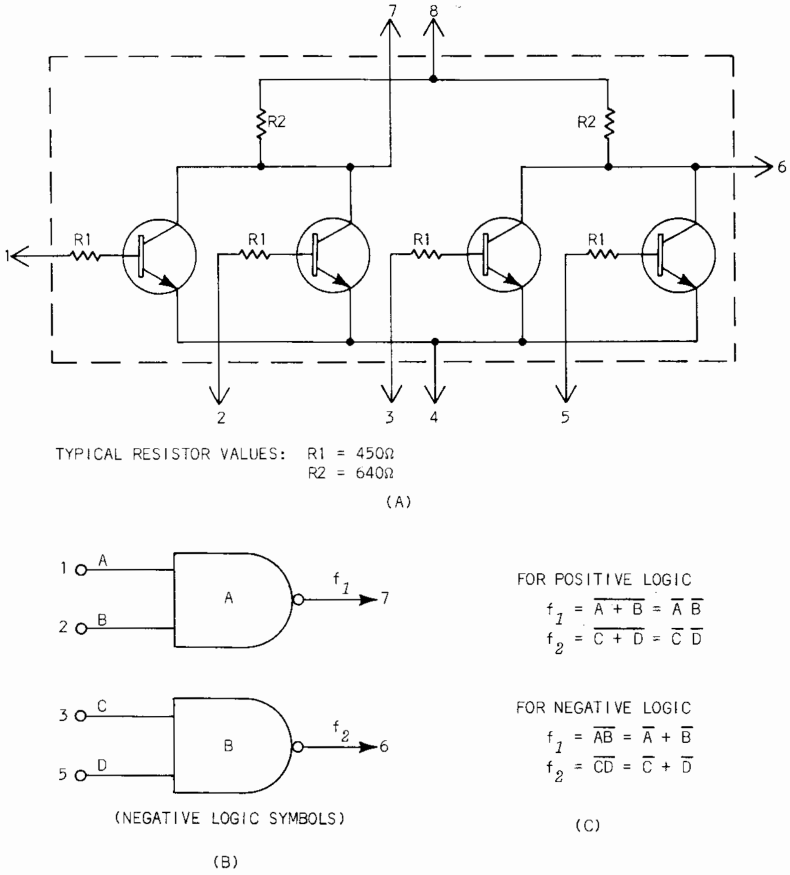

7
8
R2
R2
6
1
R1
R1
R1
R1
2
3
4
5
TYPICAL RESISTOR VALUES: R1 = 450Ω
R2 = 640Ω
(A)
1 A
2 B
A
f_1
7
3 C
5 D
B
f_2
6
(NEGATIVE LOGIC SYMBOLS)
(B)
FOR POSITIVE LOGIC
$f_1 = \overline{A + B} = \overline{A}\,\overline{B}$
$f_2 = \overline{C + D} = \overline{C}\,\overline{D}$
FOR NEGATIVE LOGIC
$f_1 = \overline{AB} = \overline{A} + \overline{B}$
$f_2 = \overline{CD} = \overline{C} + \overline{D}$
(C)

Fig. 6-1. 914 dual two-input gate.

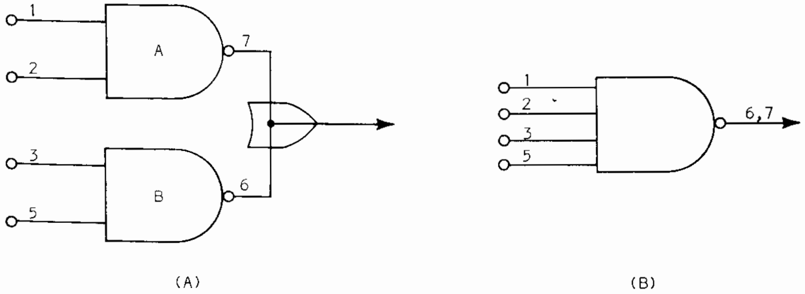


Fig. 6-2. The 914 connected as a four-input NAND gate.

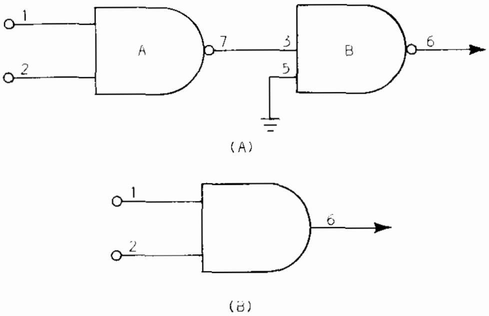


Fig. 6-3. The 914 connected for AND operation.

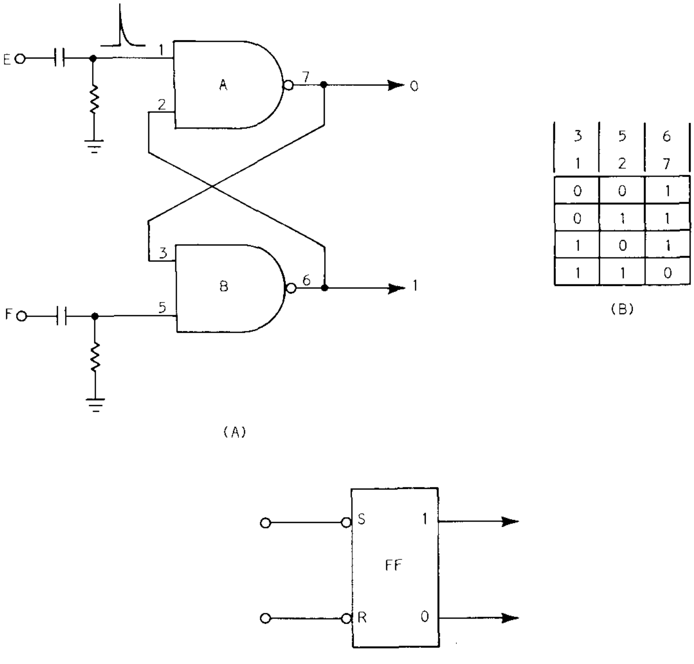
914 as AND gate

Should an AND function be required, a 914 is sometimes operated as shown in Fig. 6-3A. The output of gate A is $\overline{AB}$. Gate B with pin 5 grounded inverts. Therefore, $\overline{AB}$ inverted becomes $\overline{\overline{AB}} = AB$. On certain logic diagrams the whole operation may appear with a single AND-gate symbol as in Fig. 6-3B.

914 as FF

The dual two-input gate elements may be cross-connected to form FF's. A FF formed by cross-connecting a 914 appears in Fig. 6-4A. To understand the operation of the circuit, first consider the fact that pins 1 and 5 are quiescently 0 V (true). Assume that pin 2 is also true. NAND-gate A output, pin 7, is therefore false. The false at pin 7 connects to pin 3 of gate B. Gate B output is therefore true. This true connects back to pin 2 proving that the original assumption for pin 2 is valid. As long as no input signal appears at inputs E and F the circuit is stable in this condition with output 1 true and output 0 false.

Because of the series capacitors, an input to E or F must be a step function. For this circuit, an input signal could be either a negative step (change from false to true) or a positive step (change from true to false). At inputs E or F, a negative step is not recognized since pins 1 and 5 are quiescently true. Assume a positive step input to E. Pin 1 goes momentarily false and output pin 7 goes true. Pin 3 goes true, pin 5 is already true, therefore pin 6 goes false. This false level couples to pin 2. After the AC-coupled false signal to pin 1 decays, pin 6 remains false. The FF has changed states. This is also a stable condition.



(A)

3	5	6
1	2	7
0	0	1
0	1	1
1	0	1
1	1	0

(B)

(C)

Fig. 6-4. The 914 connected as an RS FF.

Any additional input signals to E cause no further switching. A positive edge at F, however, switches the flipflop back to the original condition. The reader may trace the levels through to determine the sequence of operation.

This circuit is a form of RS FF. Call pin 6 the 1 output, then input E is the Set input and F the Reset input. The circuit could be represented by the symbol of Fig. 6-4C. However, in Tektronix logic diagrams the circuit most often appears as in Fig. 6-4A. It is important to recall that an RS flipflop has an ambiguous state. If inputs E and F have simultaneous positive edge signals the state of the FF cannot be predicted.

On Tektronix logic diagrams for the Type 230 all IC's are identified as MXXXX (M1801, M2302, etc.) For this reason most IC's found on diagrams in this book use MXXXX as a reference number. For the Type 240 and all later digital instruments the logic diagrams will use "UXXXX" instead of MXXXX.

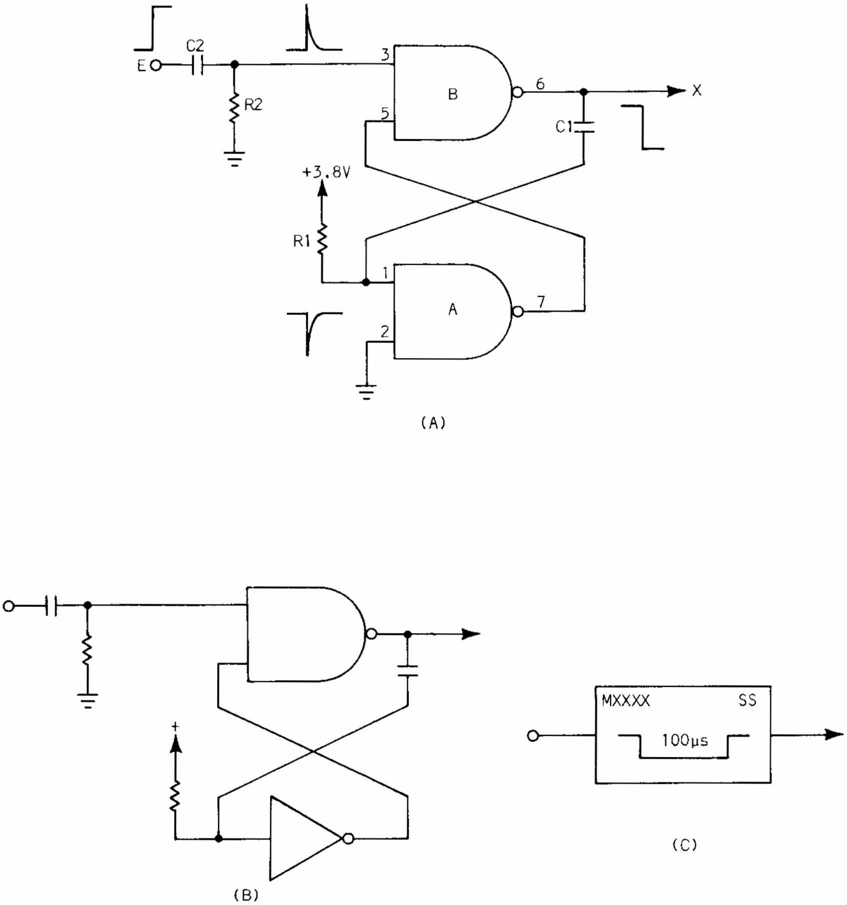


Fig. 6-5. The 914 operated as a single-shot circuit.

single shot

Occasionally the output signal of a logic circuit has insufficient pulse width. A circuit which can "stretch" such a pulse appears in Fig. 6-5A. This circuit is a single shot, also called a one shot. Pin 3 is quiescently true. Pin 1 is held false by the return through R1 to +3.8 V. Pin 7 is quiescently true holding pin 5 true. Gate B with both inputs true has output pin 6 false. The circuit is stable in this condition.

A positive edge (true to false) at input E drives pin 3 momentarily false; pin 6 goes true. The negative change at pin 6 from false to true AC-couples through C1 to pin 1. Pin 1, momentarily true, causes pin 7 to go false. This false level coupled to pin 5 reinforces the false level at pin 3. The signal at pin 3 goes true after a time determined by R1-C1. However, pin 5 remains false so pin 6 does not change. The negative edge at pin 1 remains true for a period of time determined by R1-C1. When pin 1 goes false again the circuit resets to the original state.

The output pulse width is determined by the R1-C1 time constant. For a typical circuit of this type with R1 = 3.3 kΩ and C1 = 0.1 μF, the output pulse width is about 150 μs. The original input signal was a positive pulse while the output is negative. If a negative pulse cannot be tolerated then the output could be taken from pin 7 of the 914.

NAND gate A actually operates as an inverter, so this circuit frequently appears as in Fig. 6-5B. Tektronix logic diagrams use the symbol of 6-5C. The letters SS stand for single shot. The sketched waveform shows output-pulse duration and polarity.

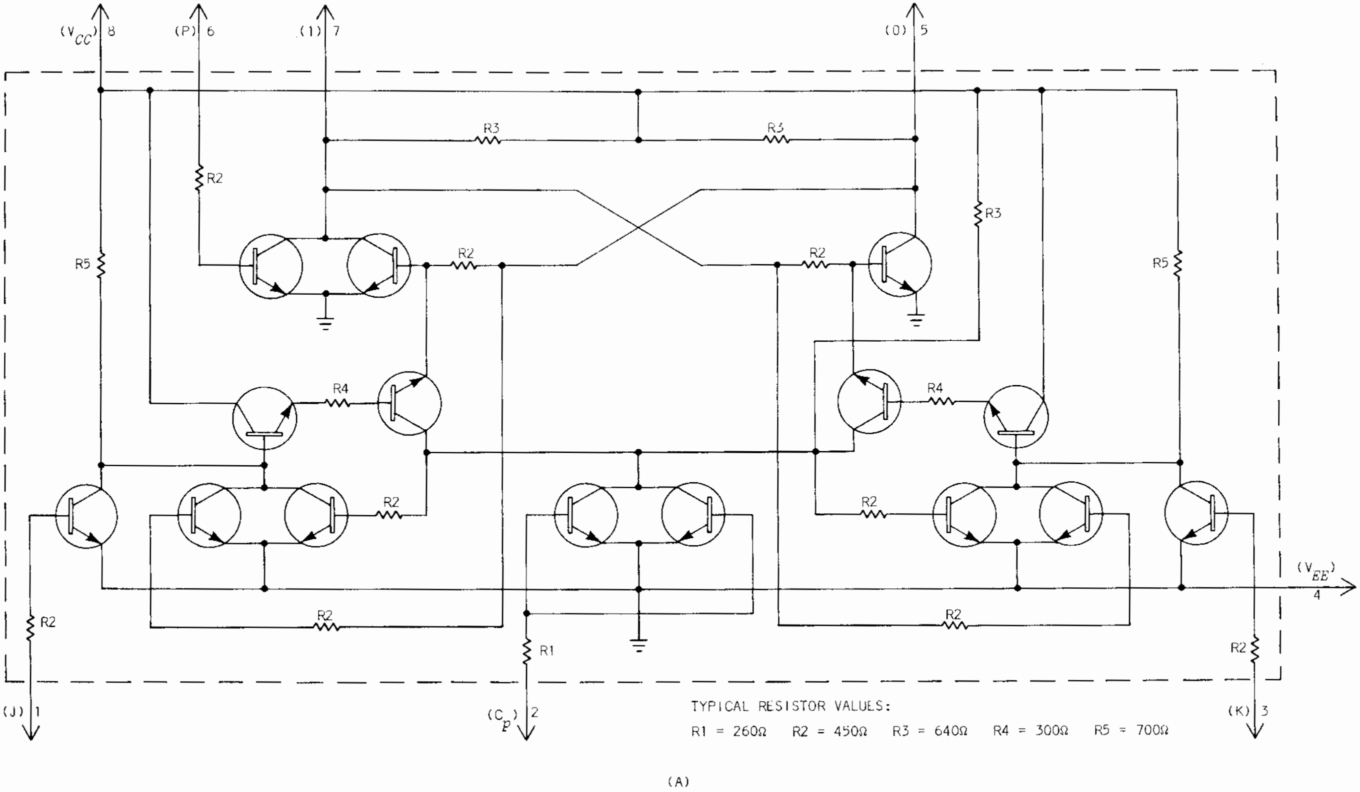
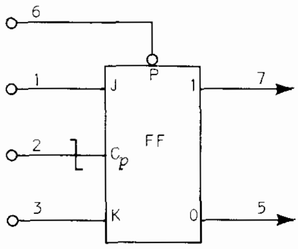


Fig. 6-6. Type 923 JK flipflop.



(B)

TRUTH TABLE

t = N		t = N+1	
J	K	1	0
0	0	NO CHANGE	
0	1	0	1
1	0	1	0
1	1	N+1	$\overline{N+1}$

0 IS MORE POSITIVE THAN 1

N = BIT VALUE BEFORE NEGATIVE CLOCK PULSE

N+1 = BIT VALUE AFTER CLOCK PULSE

(C)

Fig. 6-6. Type 923 JK flipflop.

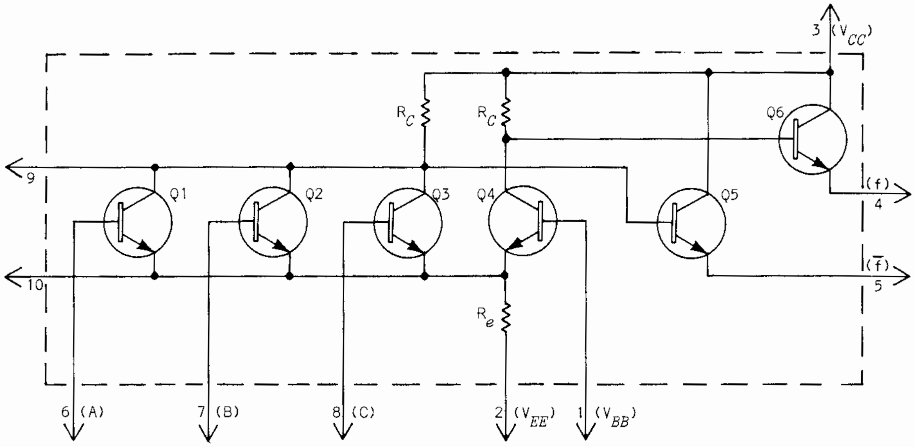
Fairchild
923

Another type of Fairchild integrated circuit widely used in Tektronix instruments is the Type 923 which is a clocked JK FF. The schematic diagram for the actual circuitry on the chip is shown in Fig. 6-6A. The schematic symbol used by Tektronix is shown in Fig. 6-6B. The 923 has J, K, C_p (Clock) and P (Preset) inputs. The outputs are identified as 1 and 0 outputs, respectively. Power-supply requirements are the same as required to operate the Fairchild Type 914, +3.8 V.

923 as
clocked
JK FF

The symbol shown connected to the C_p or clock input indicates that the C_p input is actuated by a negative edge. That is, a signal must be at the false level and move to the true level in negative logic to actuate the FF. It is very important to recall that with a clocked JK FF both J and K inputs can be made any combination of true and false without affecting the output of the FF until a clock pulse appears at the clock input. A truth table appears in Fig. 6-6C.

Several new terms have been introduced within the truth table. The combination of the J and K inputs are shown in 0-1 combinations in the column labeled t = N. Interpret t = N to mean bit value before a negative-going clock pulse appears at C_p. The second column lists the output conditions for t = N + 1 where N + 1 means the bit value after a clock pulse appears.

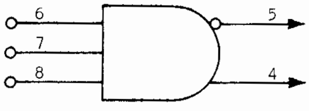


(A) SCHEMATIC

6	7	8	5	4
A	B	C	$\overline{f}$	f
0	0	0	1	0
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	0	1

0 IS MORE POSITIVE THAN 1

(B) TRUTH TABLE



(C) LOGIC SYMBOL

Fig. 6-7. MC 357 3-input AND/NAND gate.

Examining the first line of the truth table, we find that J and K inputs are both indicated as being false. For a JK FF, if both inputs are held false, the FF is inhibited; therefore, in the output columns we show the words "no change". This tells us that the FF will hold its present state, whatever it is. If J is false, and K is true, with a clock pulse, the 1 output goes to 0, the 0 output goes to 1. Just the opposite occurs if J is true and K is false.

If J and K are both true, this is equivalent to grounding both J and K inputs. The clock pulse in this case switches the FF from its present condition to the opposite condition. Since the present condition may be either of the two possible states we use the symbol N + 1, meaning the FF will switch to the opposite state.

The Preset input is a special purpose input which allows placing the JK FF in a known state with a single pulse. The P input responds only to the positive edge of a signal. The signal must go from true to false. The input is insensitive to a negative edge. A positive-edge signal at P forces the 1 output to a logical 1. Preset is a priority input. When actuated, P overrides the J, K, and C_p inputs.

emitter-coupled logic

Motorola MC 357

Motorola labels the IC's used in Tektronix instruments as MECL. These initials stand for Motorola Emitter-Coupled Logic. Emitter-coupled logic is a form of TTL logic which was discussed in Chapter 5. Fig. 6-7A shows the schematic diagram for a Motorola Type MC-357 three-input AND/NAND OR/NOR gate. The three input-transistor bases connect to pins 6, 7, and 8. The collectors of these three transistors are paralleled and connected to the base of Q5. Q5 operates as an emitter follower providing an output at pin 5. The emitters of Q1, Q2, and Q3 are common to R_e and connect to the emitter of Q4. Q4, a grounded-base amplifier, connects to the base of output emitter-follower Q6. The emitters of Q5 and Q6 connect directly to output pins 5 and 4, respectively. Each transistor requires an external-emitter load. Wherever this gate is used the load resistance is supplied.

MC 357
as AND or
NAND gate

The truth table for the MC 357 is shown in Fig. 6-7B. Examination of the truth table shows (for negative logic) when using output pin 5 a NAND gate function is performed. Output pins 4 and 5 are complementary. Using pin 4 as the output, f = ABC; using pin 5, $\overline{f} = \overline{ABC}$.

MC 357
as OR
or NOR
gate

This gate operates as either a NAND gate or an AND gate (or both) depending upon the output used. In some circuits, (Tektronix Type 230), both are used. The reader must be very careful to consider which output is used before deciding what logic function is taking place. For positive logic this gate acts as an OR/NOR gate. The MC 357 logic symbol appears in Fig. 6-7C.

Motorola 350-series IC's may require three different voltage supplies. In examining the schematic of the MC 357 note that pin 3 is connected to V_{cc}, the symbol for collector voltage supply. Pin 1 of the chip is connected to V_{bb}, a bias-supply input. Pin 2, V_{ee}, emitter-supply voltage, is the third power supply input.

Manufacturer's specifications state V_{cc} = 0 V, V_{bb} = -1.158 V, V_{ee} = -5.2 V. These conditions give output levels of -1.55 V low and -0.75 V high. Since Tektronix instruments require a low level of 0 V the 357 is operated with V_{cc} +1.75 V, V_{ee} -3.5 V, and V_{bb} +0.65 V. This yields output levels of 0 V low and +0.8 V high. The MC 350-series IC's are available in 10-lead metal packages of TO-5 size. Propagation time is about 6 ns. Fanout is good, the gate output can drive up to 26 other MC 350-series inputs.

Examining the MC 357 again, the collectors of Q1, Q2 and Q3 are connected. This connection is brought out to pin 9. The emitter bus is brought out through pin 10. Adding discrete transistors between pins 9 and 10 enables gate expansion.

That is, the base of an external transistor, with collector connected to pin 9 and emitter to pin 10, becomes an additional input to the gate. The truth table would be modified to that of a 4-input gate.

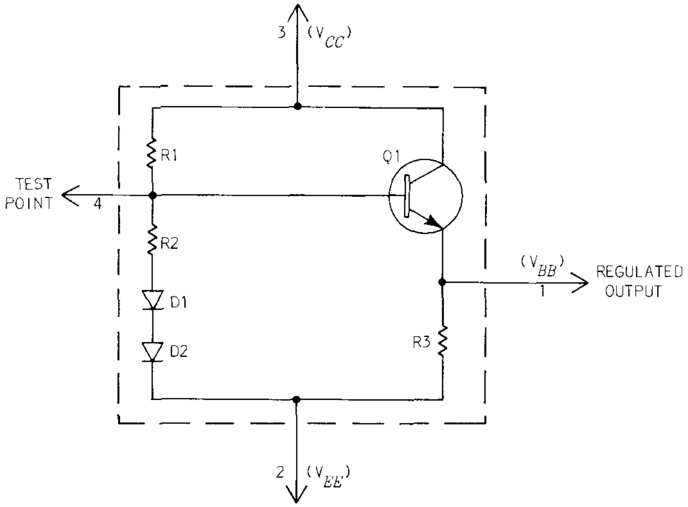


Fig. 6-8. MC 354 regulator.

Motorola MC 354 regulator

Fig. 6-8 shows the schematic diagram for the Motorola Type MC 354 bias driver. The V_{bb} input on the Motorola 350-series gates requires a stabilized voltage. Since the logic gates require a well regulated voltage, the MC 354 has been manufactured to make a stable voltage source readily available. Examining the schematic of the chip, a transistor (Q1) is located on the chip. The collector connects to V_{cc}, and its emitter-load is R3. Q1 operates as an emitter follower. The base voltage of Q1 is determined by a resistive divider, R1 and R2. In series with the R2, D1 and D2 are present to provide temperature compensation. The diodes insure stable and reliable operation of the device over the temperature range of 0°C to +75°C. With 5.2 V ±20% applied to pin 3, the output pin 1 will be approximately +1.15 V. Since the IC performs no logic function, no logic symbol for the device is used.

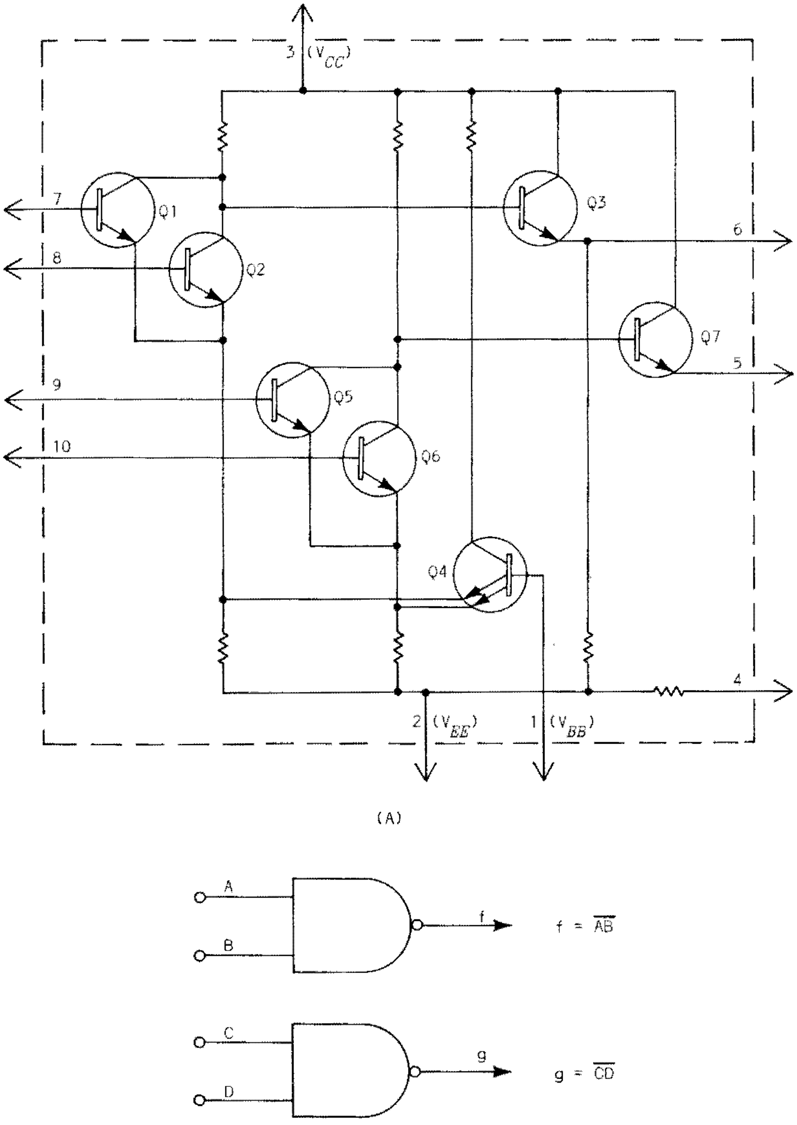
Motorola MC 360 NAND gate

In Fig. 6-9A a dual two-input NAND gate Motorola Type MC 360 is shown. There are two independent gates present in this circuit. Input pins 7 and 8 connect to the Q1 and Q2 bases respectively. The paralleled collectors of Q1 and Q2 connect to Q3 base. Q3 functions as an emitter follower to output-pin 6. Three other transistors, Q5, Q6, and Q7 operate in a similar fashion. Q1 and Q2 have their emitters parallel-connected and tie to one emitter of a special transistor Q4 made with two separate emitter-base junctions.

Connecting Q1 and Q2 emitters to Q4 provides a temperature-stabilized bias voltage to Q1 and Q2. The other emitter of Q4 connects to the other two-input gate configuration on the chip. The base of Q4 is brought out to pin 1 (V_{bb}) and typically connects to the output of the previously mentioned Type MC 354.

The logic symbol of the MC 360 appears in Fig. 6-9B. Since the same symbol is used for both the MC 360 and Fairchild µL 914 dual NAND gates it is impossible to know which type is used. Yet, it is important sometimes to know which IC is used at that point in a circuit. The µL 914 may be operated as a four-input NAND gate by connecting the outputs in parallel. If the outputs of the MC 354 are connected together a four-input NAND gate does *not* result.

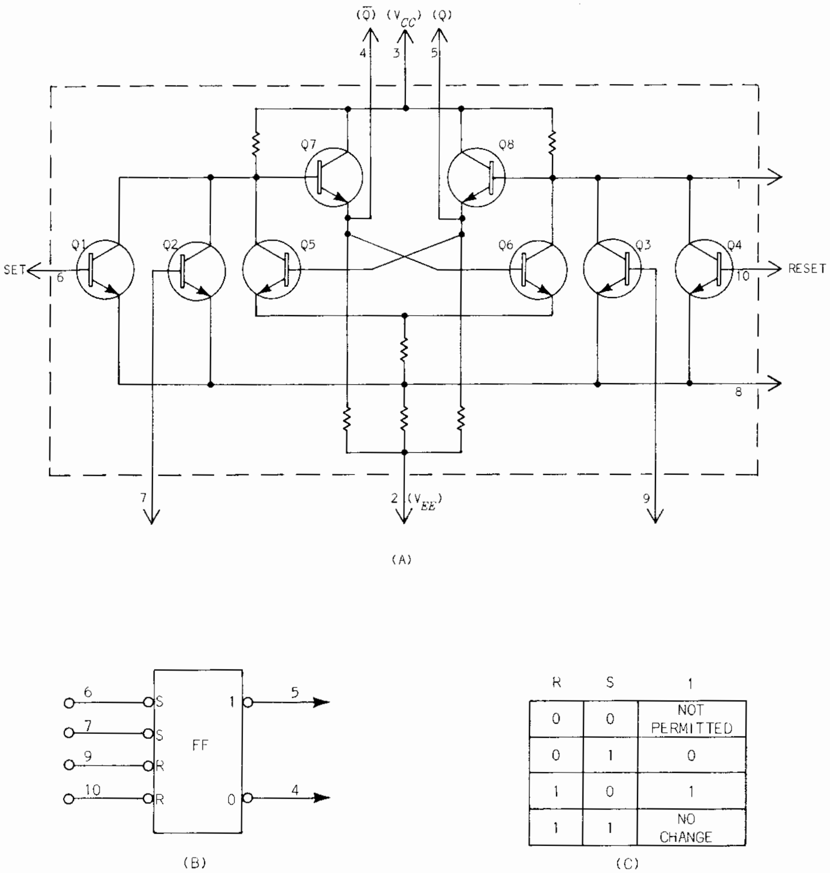
Look at output pins 5 and 6 of Fig. 6-9A. If pins 5 and 6 are paralleled, two NPN emitter followers are connected together. If either Q3 or Q7 has a high-level (false) output the composite output is high. For negative logic this is not a phantom OR. It is a phantom AND connection. The complete expression takes the output of each NAND gate and AND's the results $X = \overline{AB} \cdot \overline{CD}$. On logic diagrams the phantom AND symbol appears at the junction of the two NAND gates.



(A)

(B)

Fig. 6-9. MC 360 dual two-input NAND gate.



(A)

(B)

R	S	1
0	0	NOT PERMITTED
0	1	0
1	0	1
1	1	NO CHANGE

(C)

Fig. 6-10. MC 352 RS FF.

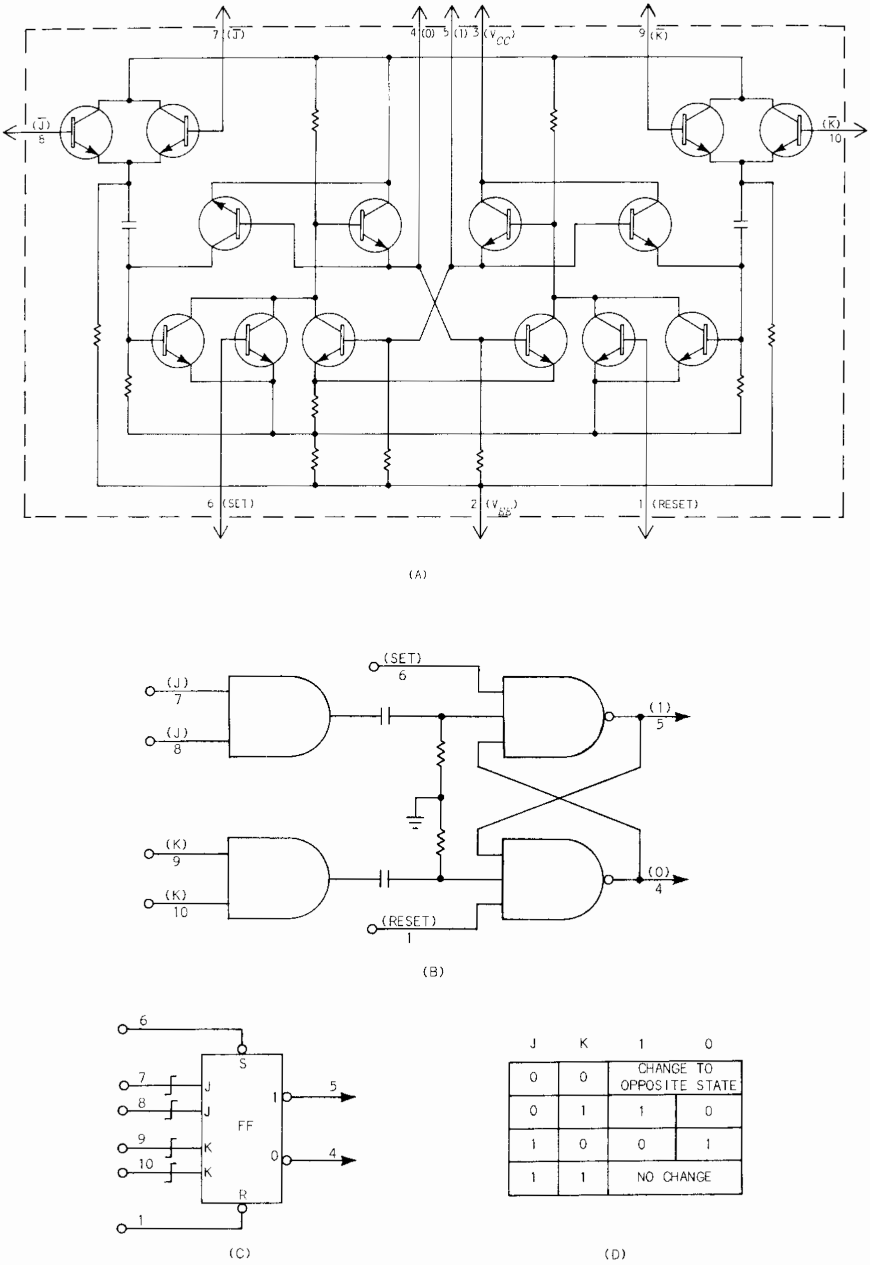
Motorola MC 352 RS FF

The MC 352 flipflop schematic is shown in Fig. 6-10A. Eight transistors are present on the chip. Transistors Q1 and Q2 form the Set inputs; Q3 and Q4 form the Reset inputs. Transistors Q5 and Q6 form the actual FF. Transistors Q7 and Q8 are emitter followers from which the outputs 1 and 0 are taken. In order to change the state of the FF via the Set input, Q1 or Q2 must turn on.

To turn either on requires a positive signal level. In a negative logic environment a false level actuates the FF. This also applies to the Reset input. When Q1 turns on, its collector pulls down, pulling down the base of Q7. Q7, acting as an emitter follower, pulls down on the base of Q6, turning it off. Q6 collector goes high. Q8 emitter also goes high which means that a false input at the Set input causes the 1 output to go false.

This circuit was originally designed to act as an RS FF for use with a positive-logic assignment. Here, the Set input is a false level and in turn the 1 output goes to a false level. For a negative logic assignment therefore, this FF might be called a $\overline{\text{Set}}$ $\overline{\text{Reset}}$ FF. A set input being a false level could be called a $\overline{\text{Set}}$ signal. The one output could be called a $\overline{\text{one}}$ output. Using positive logic FF's in a negative logic environment complicates the naming of the various inputs and outputs. We only say here that a $\overline{\text{Set}}$ pulse causes the FF to go to a $\overline{\text{one}}$ state. In the same manner we could say that a $\overline{\text{Reset}}$ input causes the FF to go to a $\overline{\text{zero}}$ level. In all cases, of course, the NOT inverts the signal level. The Tektronix logic symbol for this FF is shown in Fig. 6-10B. A truth table for the FF is shown in Fig. 6-10C.

For this particular device two high inputs are not allowed. As is usual there is an ambiguous state for the FF. The MC 352 is compatible with all other Motorola MC 350 series logic devices. It has an average propagation delay of 10 ns. Power requirements are exactly the same as for the MC 357.



J	K	1	0
0	0	CHANGE TO OPPOSITE STATE	
0	1	1	0
1	0	0	1
1	1	NO CHANGE	

Fig. 6-11. MC 358 JK FF.

Motorola
MC 358
JK FF

The Motorola MC 358 is a JK FF designed for use with all other series MC 350 integrated circuits. The device is packaged in a TO-5 configuration with 10 leads. The case is metal. This JK FF is capable of relatively high-speed operation; it can be clocked at a rate of approximately 30 megahertz. A schematic for the chip is shown in Fig. 6-11A. This FF has J and K inputs as well as Set and Reset inputs. The device is designed to perform as an RS FF, a JK FF, or a clocked JK FF.

A logic diagram for the circuitry *on the chip* appears in Fig. 6-11B. Any unused inputs can be left floating or grounded.

As an RS FF the MC 358 would operate in the same manner as the MC 352.

The JK FF logic diagram symbol is shown in Fig. 6-11C. The edge symbols at J and K inputs indicate that they respond to a false edge. A truth table for this symbol appears in Fig. 6-11D. If both J and K go false the FF toggles from its present state. The reader may trace these levels through the diagram of Fig. 6-11B. The fact that J and K inputs are duplicated permits a form of inhibiting logic. If pin 7 is held false, pin 8 is inhibited. That is, a positive edge on pin 8 has no effect. On the other hand, if pin 7 is held true then pin 8 can switch the multi. Thus, in certain circuits a J or K input is inhibited by the other input connection. In other circuits it may be required that the FF be placed in a known condition independently of the J and K inputs. Either the Set or Reset inputs may be used depending on the desired FF state. By connecting pins 8 and 9 together a clocked JK FF results. The C_p input thus formed is sensitive only to a positive edge signal. The logic diagram symbol is shown in Fig. 6-12.

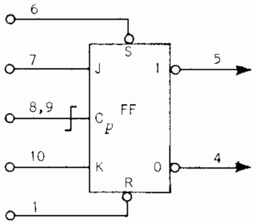


Fig. 6-12. MC 358 connected for clocked-mode oœration.

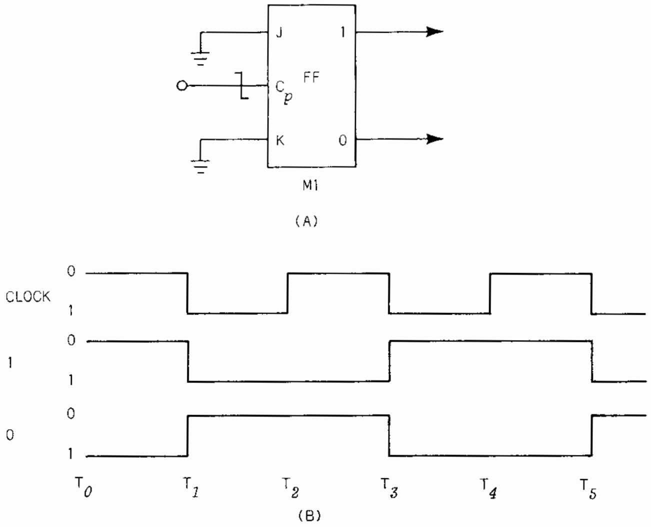


Fig. 7-1. ÷2 circuit.

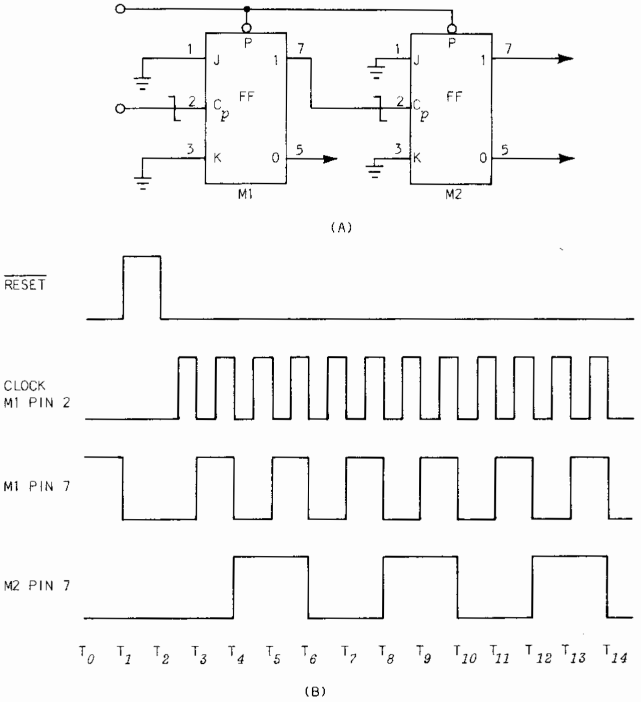


Fig. 7-2. ÷4 circuit.

7

COUNTING CIRCUITS

÷2 Circuits which can count are frequently required in digital instruments. Events to be counted are presented to the counting circuit in the form of a pulse train, one pulse per event. It is possible to construct circuits which divide the input signal's frequency by any desired ratio. For example, a circuit which has one output pulse for every two input pulses is a divide-by-two circuit.

Any flipflop which can act as a Toggle FF is a ÷2 circuit. The clocked JK FF with J and K inputs held true operates in a toggle mode. See Fig. 7-1A. A Fairchild Type 923 JK FF is shown with J and K both true. A waveform ladder diagram appears in Fig. 7-1B. The clock input is a pulse train representing the events to be divided-by-two. The input responds only to a negative edge.

At time T_1, the clock signal changes from 0 to 1 and the negative edge toggles M1. The 1 output goes from 0 to 1 and the 0 output goes to 0.

At T_2 clock goes positive but M1 does not respond.

At T_3 clock again goes negative and M1 toggles.

At T_4 clock goes positive and again M1 does not respond.

At T_5 clock goes negative and M1 toggles.

At this point the reader can see that while the clock signal has completed two cycles the 1 output has completed one cycle. This is a ÷2 action.

÷4 By connecting two ÷2 circuits in cascade, a ÷4 circuit results. See Fig. 7-2A. Two Fairchild Type 923's are shown. The Preset inputs are shown. In some counting circuits it is necessary to provide a means for returning the circuit to a known state. This is here accomplished with the Preset input. A waveform ladder diagram appears in Fig. 7-2B.

At time T_0 M1 is high, M2 pin 7 is low.

At time T_1 a positive pulse called $\overline{\text{Reset}}$ appears and FF's M1 and M2 reset to 1.

At Tektronix a waveform which is logically false at the time it accomplishes its object is identified by the bar. $\overline{\text{Reset}}$ is read aloud as Reset Not.

At some later time after T_2 a series of clock pulses arrive at M1 pin 2. The negative edge of each pulse toggles M1 at times T_3, T_4, T_5, etc. M1 pin 7 switches negative at every other clock pulse, T_4, T_6, T_8, etc. Each negative edge constitutes a carry to M2 pin 2.

M2 divides by two and its output pin 7 has one complete output cycle per four input clock cycles to M1.

÷8

÷16

At this point it is apparent that by using toggle FF's any dividing ratio which is a whole number power of 2 can be obtained. Should a ÷8 ratio be required an additional FF could be connected to the output of M2 of Fig. 7-2A. Should a ÷16 ratio be desired, yet an additional FF could be added.

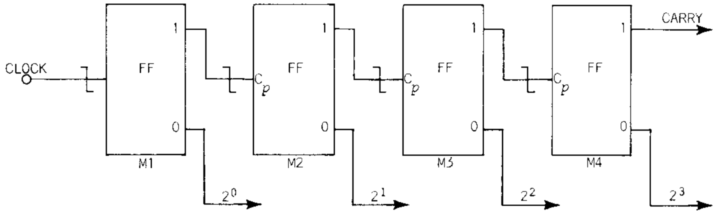
readout

Certain counting circuits are required to *count* for some period of time and then display the results. Some means of *readout* must be provided for the total of the count to be known. Consider the circuit of Fig. 7-2A and assume that the count period ended at time T_5 after three clock pulses. With no further clock pulses M1 and M2 hold the condition existing at T_5 until $\overline{\text{Reset}}$ appears again.

M1 is in the 0 state; pin 7 is high. M2 is also in the 0 state. This means that the 0 output, pin 5, is at the 1 state for M1 and M2. Assume a circuit is available which can recognize the levels at the 0 output pins. Such a circuit would see that the counter has a binary number 11 (3_{10}) stored. A problem is which of the output 1's represent 2^0 and which 2^1. See Fig. 7-3A. To make a binary count from 0 to 4 the column labeled 2^0 changes states with *each* count. Restudy Fig. 7-2A and see that M1 changes states with each pulse to be counted. This leads us to assign the value 2^0 to M1 and 2^1 to M2.

DECIMAL	BINARY		
	2^2	2^1	2^0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0

(A)



(B)

Fig. 7-3. ÷16 circuit.

Conventional diagrams place the input on the left and output on the right. For a divide-by-16 circuit this leads to the situation shown in Fig. 7-3B. The 2^0 output is on the left. The 2^3 output is on the right. *When writing a binary number the least significant number is on the right, the reverse of readout on the diagram.* The reader should be alert to this condition and be able to identify the binary weighting of each FF. The conventional way of identifying FF's in a counter calls M1 the 1 binary, M2 the 2 binary, M3 the 4 binary, and M4 the 8 binary. This may be reconciled to the powers of two by recalling that $2^0 = 1$, $2^1 = 2$, $2^2 = 4$, and $2^3 = 8$. A readout of Fig. 7-3B would be in the form of a four-digit binary number. The value could be any number between 0000_2 and 1111_2. 1111_2 is 15_{10}. The highest number that can be represented in the ÷16 circuit is 15_{10}. The reader may determine for himself that the highest number for other counting circuits is always one less than the divide ratio.

Simple toggle FF circuits may be used if a counter is required which divides by a power of 2. If any other number ratio is required the designer is forced to use other more specialized types of FF's. The basic scheme to solve the problem utilizes binary FF's and feedback systems.

÷3

Consider a ÷3 ratio. Such a ratio is not a whole number power of two. Since 3 is greater than 2^1 but less than 2^2 two binary stages are required. The circuit can be made to ÷3 if there is some way for the basic ÷4 circuit to go through a complete cycle but with only three input pulses. Many schemes have been developed to perform this function. One example is shown in Fig. 7-4.

Here the 0 output of M1 drives M2 and the 0 output of M2 is AC-coupled back to the Preset input of M1. At T_1, M1 toggles and pin 5 generates a negative edge. M2 also toggles. The negative edge of M2 pin 5 is differentiated and fed as a negative spike to M1 Preset. Preset does not respond to a negative signal.

At T_2, M1 toggles again and the 0 output generates a positive edge which does not toggle M2.

At T_3, M1 toggles and the 0 output toggles M2. The positive edge, differentiated by R1C1 to a positive spike, presets M1. M1 pin 5 goes false again. The width of the waveform at M1 pin 5 will be determined by the response times of the FF's. Here it is exaggerated for clarity.

At T_3, M2 pin 7 goes true. This negative signal is the output of the circuit. The output goes true once for every three input signals. This satisfies the ÷3 requirement. No readout circuitry is shown because this circuit operates in a continuous fashion. No means is provided for periodically resetting the circuit.

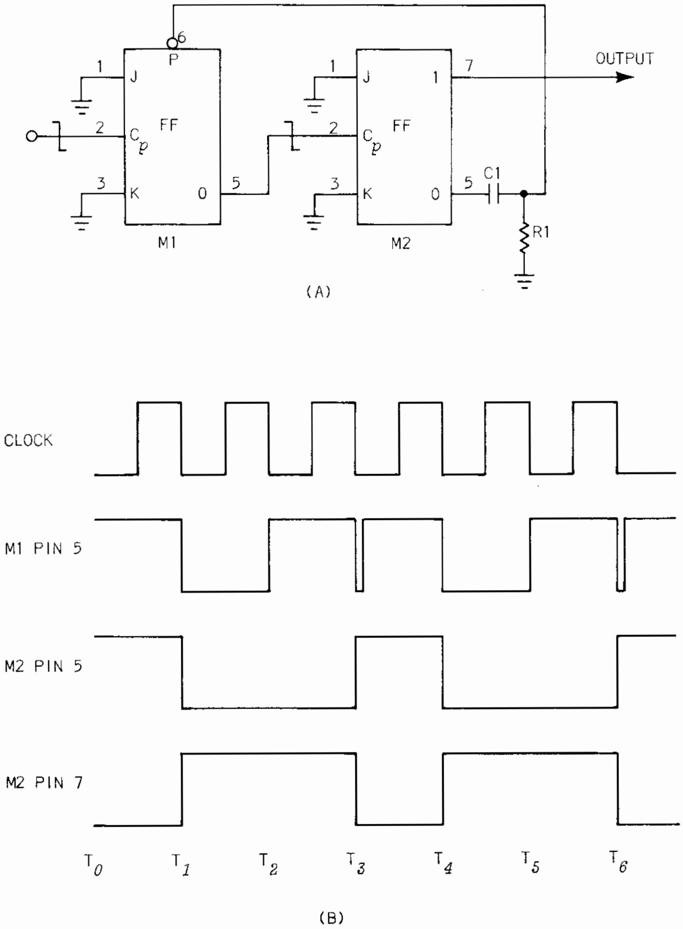
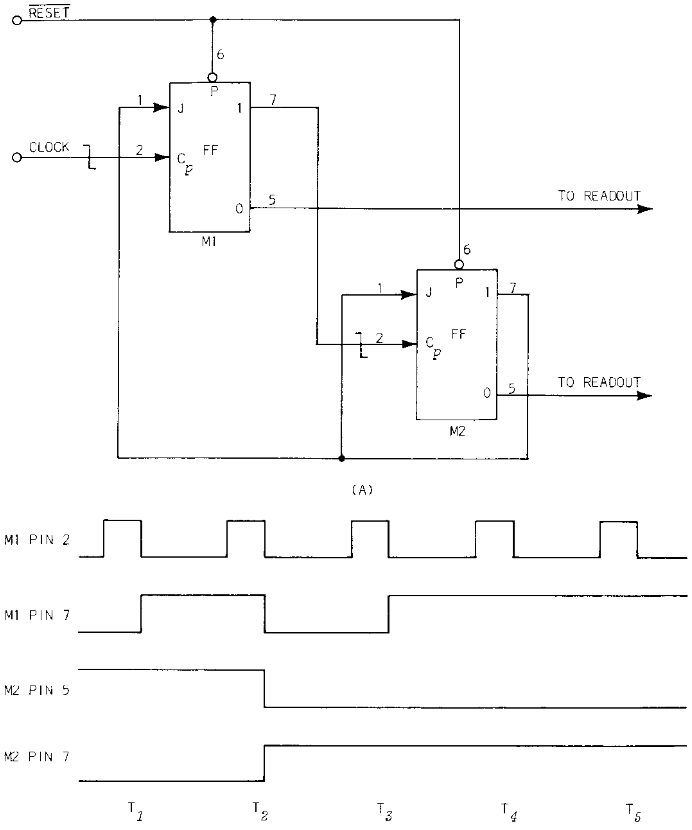


Fig. 7-4. ÷3 circuit.



(A)

(B)

CLOCK	M2 PIN 5	M1 PIN 5
0	0	0
1	0	1
2	1	0
3	1	1
4	1	1
5	1	1

(C)

Fig. 7-5. ÷3 circuit.

Another form of ÷3 circuit appears in Fig. 7-5. This circuit counts to three and then locks itself up. Once locked up a $\overline{\text{Reset}}$ pulse is required to unlock the circuit. Work your way through the sequence of the circuit with the aid of the waveform ladder diagram of Fig. 7-5B. At T_3 the false level at M2 pin 7 coupled back to both J inputs locks the FF's with pin 7 high, pin 5 low. The only method of unlocking is the arrival of a new $\overline{\text{Reset}}$ pulse. Any succeeding clock pulses are ignored until the next $\overline{\text{Reset}}$.

The truth table (Fig. 7-5C) shows that a readout circuit may read the count state of the circuit in BCD 8, 4, 2, 1. M2 pin 5 is the 2^1 output while M1 pin 5 is the 2^0 output.

Clocked SET-RESET FF's connected as dividers represent some of the more difficult concepts in counting circuits. The difficulty is that the FF's respond to positive edge signals.

÷5 Three ÷2 circuits may be connected to divide by five as shown in Fig. 7-6A. Assume the clock input is 0 and all 1 outputs are 0's (See Fig. 7-6B, T_1).

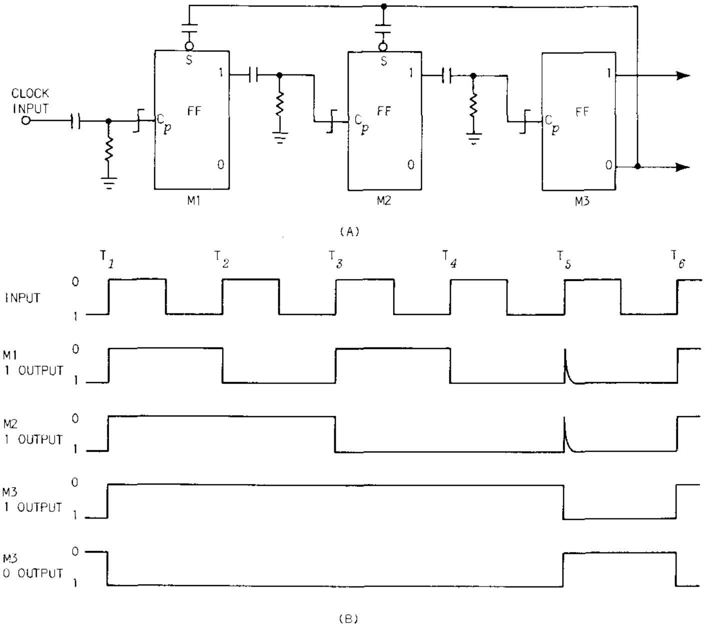


Fig. 7-6. ÷5 circuit.

M1 and M2 form a divide-by-4 circuit as shown in Fig. 7-6B from T_1 to T_5. When the input changes at T_5, M1 and M2 switch. M3 receives an input and it switches. Its 1 output changes from 0 to 1. Its 0 output changes from 1 to 0 and this output is fed back to the Set inputs of both M1 and M2, causing them to switch.

This feedback cancels the fourth count of the divide-by-four section of the circuit. On the next input leading edge all three FF's will switch, obtaining from input to output a divide-by-five function.

Fig. 7-7A shows a different method of obtaining a divide-by-five function.

Assume that just prior to T_1 (Fig. 7-7B), the clock input is 1 and all 1 outputs are 1's. At T_1 the input steps from 1 to 0 and M1 switches, M2 switches and M3 switches.

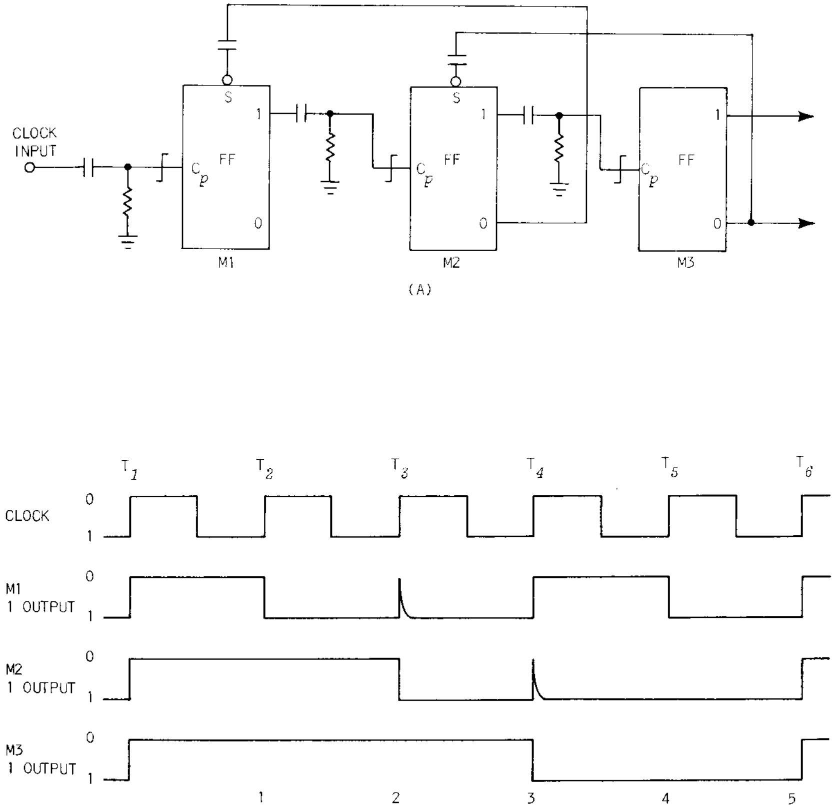
At T_2, M1 switches but nothing else happens.

At T_3, M1 switches and M2 switches, feeding back its 0 output (the inverse of M2 output signal shown in Fig. 7-7B) to the Preset input of M1 which changes M1's state to a 1 output.

At T_4, M1 switches, M2 switches and M3 switches, feeding its 0 output back to the Preset input of M2, changing M2's output back to 1.

At T_5, M1 switches but nothing else happens. At T_6, M1, M2 and M3 switch and a divide-by-five function has been performed.

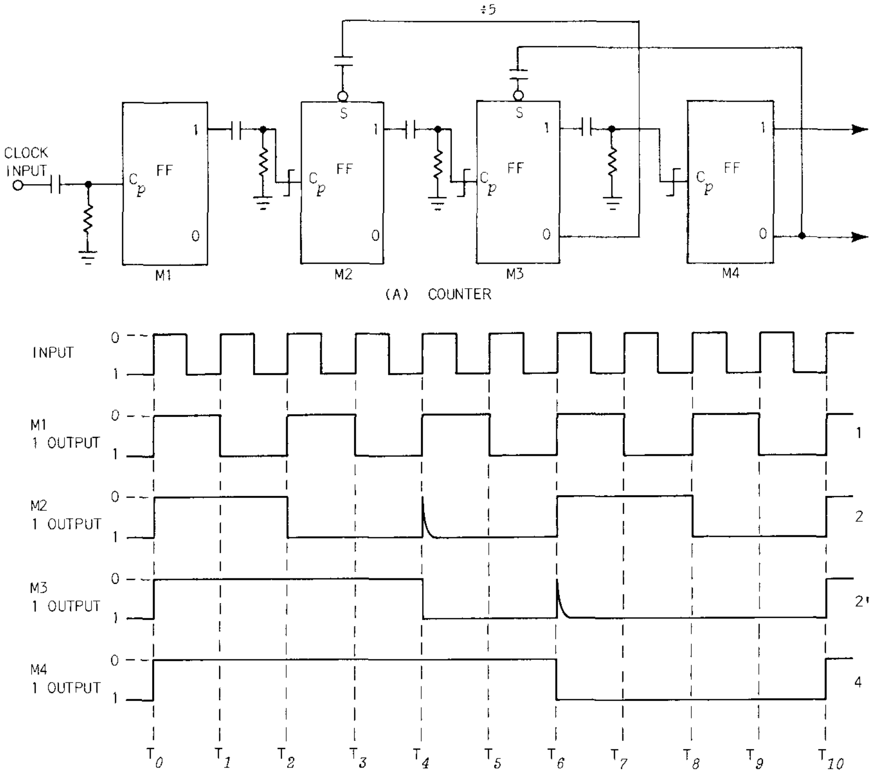
÷10 Either of the two divide-by-five circuits presented may be combined with an independent divide by two circuit to form a divide-by-ten circuit.



(A)

(B)

Fig. 7-7. Another ÷5 circuit.



(A) COUNTER

(B) LADDER DIAGRAM

TIME	M1		M2		M3		M4	
	0 OUTPUT	1 OUTPUT	0 OUTPUT	1 OUTPUT	0 OUTPUT	1 OUTPUT	0 OUTPUT	1 OUTPUT
T_0	0	1	0	1	0	1	0	1
T_1	1	0	0	1	0	1	0	1
T_2	0	1	1	0	0	0	0	1
T_3	1	0	1	0	0	0	0	1
T_4	0	1	1	0	1	0	0	1
T_5	1	0	1	0	1	0	0	1
T_6	0	1	0	1	1	1	1	0
T_7	1	0	0	1	1	1	1	0
T_8	0	1	1	0	1	0	1	0
T_9	1	0	1	0	1	0	1	0

(C) TRUTH TABLE

Fig. 7-8. 4, 2, 2', 1 weighted counter.

4,2,2',1 counter

Fig. 7-8A shows a ÷10 circuit. This counter uses a special binary code. Fig. 7-8B shows the input/ output waveforms for each FF and the BCD 4, 2, 2', 1 output levels for each count. Since two FF's have the numerical weighting of 2, one of them is identified by the prime symbol. This differentiates between the output of M2 and M3. After T_0 the 1 output of each FF is a logical one. This would read 1111, which is ambiguous since the count should be 0 at this time. For this example the binary count will be identified by the logic levels at the 0 output of the FF's. Hence, at T_0 the count is 0000.

At T_1, M1 toggles and the 0 output goes to 1. The count is 0001.

After T_2, M2 is at 1, M1 is at 0, and the count is 0010.

After T_3 the count is 0011.

M1 is given a unit weight of 1 and M2 a unit weight of 2.

After T_4, M1 is at 0, M2 is at 1, M3 is at 1, M4 is at 0. The count 0110. Since this is a count of 4 the unit weight of M3 is also 2. We identify M3's 2 by 2'.

After T_5, M4 and M1 are at 1, the others are at 0. The count is 1001. The unit weight of M4 is 4. The weight scheme for the circuit is 1, 2, 2', 4. Note that all numbers from 0_{10} to 9_{10} can be formed by combining these values.

After T_6 the outputs are 1100. M3 has the weight of a 2 and M4 must have a weight of 4.

7 is represented by 1101.

8 is represented by 1110.

9 is represented by 1111.

At T_{10}, notice that the circuit is back at the same condition as at T_0.

8,4,2,1
÷10
counter

Fig. 7-9A shows a JK divide-by-ten circuit, using Fairchild 923 JK FF's. A ladder diagram is shown in Fig. 7-9B. The truth table for the circuit is shown in Fig. 7-9C.

After a $\overline{\text{Reset}}$ pulse appears, the state of the FF's are as shown at T_0. The output of M4 is connected to the K input of M2, therefore, this input is a 1. The AND gate will have a 1 for an output only when both inputs are 1's, therefore, at T_0 its output is a 0. The output of the AND gate is connected to the K input of M4.

Each FF will toggle on a negative edge. M1 divides the input by two. The output of M1 is taken from the 1 output and fed to both M2 and M4.

The first negative edge seen by M2 is at T_2. Its J and K inputs are 1's and it toggles. M4 has a 0 for a K input and a 1 for a J input and the 1 output is a 1. M4 *will not toggle at* T_2. The input to M3 steps from 1 to 0 at T_2 and it does *not* toggle. The 1 outputs are as shown in the truth table.

At T_3, M1 toggles but nothing else happens.

At T_4, M1 toggles, feeding a negative edge to M2 which toggles. M2 feeds a negative edge to M3 and it toggles.

At T_5, M1 switches but nothing else happens.

At T_6, M1 switches and M2 switches.

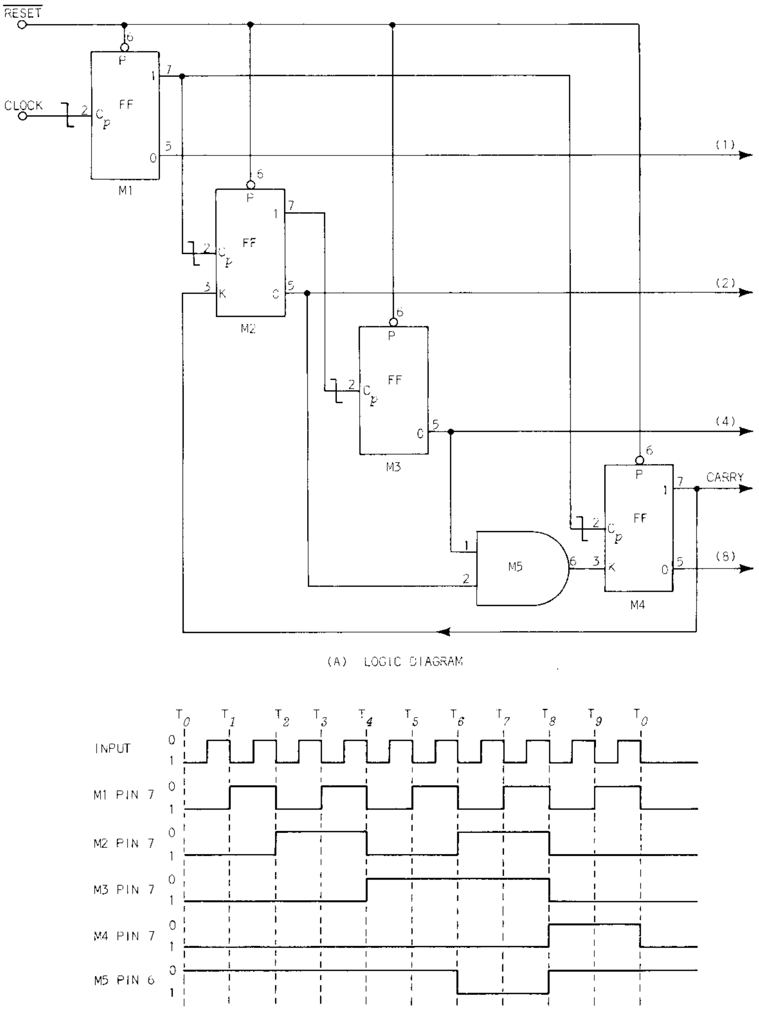
The outputs of M2 and M3 are now both 1's and the output of the AND gate M5 switches to a 1.

The important condition at this time is the K input of M4. It is now a 1 and this FF toggles on the next negative edge CP input. This occurs at T_8.

All the FF's toggle at T_8. After they have switched M2 is locked out (its K input is at 0) and M4 can only switch back with its 1 output at logical 1.

At T_9, M1 flips, M2 is locked out and the positive edge does not toggle M4.

At the second T_0, M4 toggles unlocking M2 and the situation is the same as was assumed at T_0.



(A) LOGIC DIAGRAM

(B) LADDER DIAGRAM

CLOCK	M4	M3	M2	M1	M5
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	0
10	0	0	0	0	0

(C) TRUTH TABLE

Fig. 7-9. ÷10 counter (8, 4, 2, 1 BCD).

The circuit has divided the input signal by ten and is ready to count again. Careful study of the truth table shows that at each stage of the count the number counted is shown in BCD 8, 4, 2, 1 form at the outputs of the FF's.

8,4,2,1 ÷5 or ÷10 counter

Another type of ÷10 circuit appears in Fig. 7-10A. Motorola MC 358 JK FF's are used. A special feature of the circuit is that it functions as either a ÷10 or ÷5 counter. The count ratio is determined by the logic value of the ÷5 input line. When ÷5 is logically true the circuit operates in a ÷5 mode. When ÷5 is logically false the circuit operates in ÷10 mode.

The inverter M5 is actually a clamp circuit. With the ÷5 line false the Reset input of M1 is clamped true (0 V). With the ÷5 line true the clamp is released and the level of M1 Reset is determined by the feedback line from M1 pin 5.

A ladder diagram and truth table for the circuit operating in ÷5 mode appear in Fig. 7-10B and 7-10C.

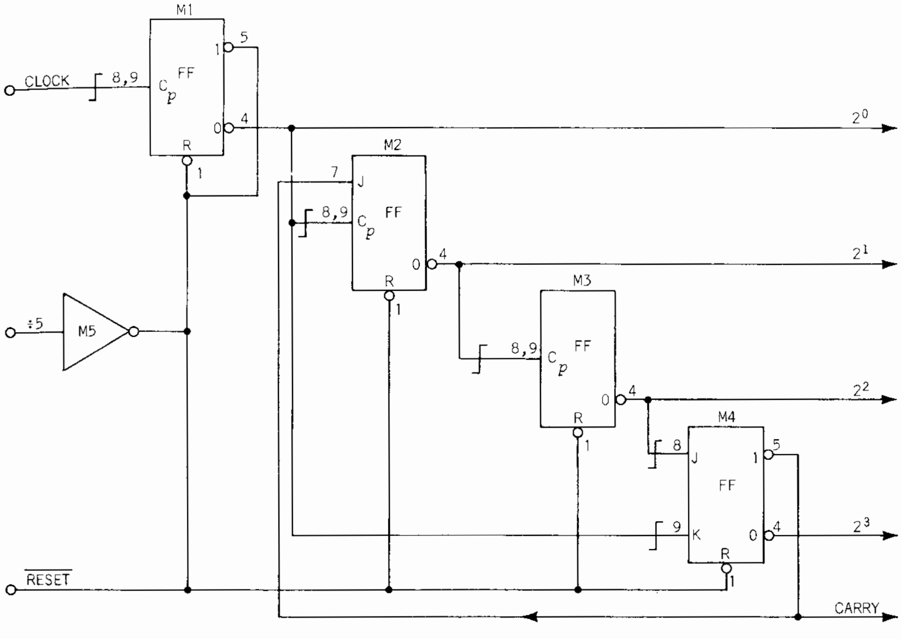
M1 operates as a monostable circuit with one output pulse generated for each clock pulse. When a clock input arrives, M1 toggles on the positive edge. Each time M1 toggles the 1 output goes from true to false. This positive level immediately resets M1 and the 1 output goes true again. M1 may be called a ÷1 circuit in this case. The output of M1 viewed from the *0* output is shown in the ladder diagram. The width of the output pulse is determined by the response time of the FF, 10 to 12 ns for the MC 358. In the drawing, the width of the pulse is exaggerated for clarity. The positive edge of each output pulse toggles M2 or M4.

At T_0, all FF's have pin 4 false.

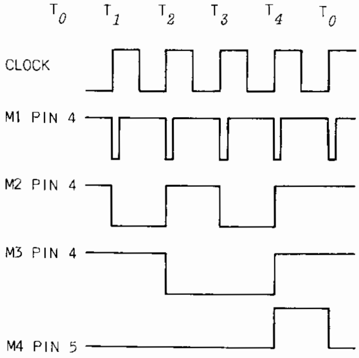
At T_1, M2 toggles, nothing else happens. M4 doesn't toggle because the J input is high.

At T_2, M2 toggles and the positive edge toggles M3.

At T_3, M2 toggles, nothing else happens.



(A) LOGIC DIAGRAM



(B) ÷5-MODE LADDER DIAGRAM

CLOCK	0 OUTPUTS PIN 4			
	M4	M3	M2	M1
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	0
4	1	0	0	0
0	0	0	0	0

(C) ÷5-MODE TRUTH TABLE

Fig. 7-10. Dual-range converter.

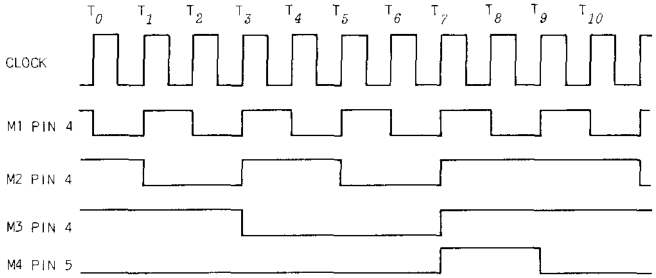
At T_4, M2, M3, and M4 toggle. The one output of M4 is now false. This high feeds back to pin 7 of M2, locking M2.

At the second T_0, M2 does not toggle. M4 has a 1 at the J input and with the arrival of the positive edge from M1, M4 switches.

M4 pin 5 goes true and pin 4 goes false. All FF's are back to the originally assumed condition.
For each 5 clock pulses, 1 carry pulse is generated at the output lead.

Next assume that the ÷5 input line is false. Inverter M5 clamps the feedback line from M1 pin 5 to pin 1 at logical 1. At this time the clamp prevents any reset action at M1 pin 1. M1 operates as a ÷2 counter. M2, M3, and M4 have already been shown to be a ÷5 counter, therefore, the entire circuit is a ÷10 counter.

The reader may follow the sequence of operation with the aid of the ladder diagram and truth table of Fig. 7-11A and 7-11B.



(A)

CLOCK	O OUTPUTS	PIN 4		
	M4	M3	M2	M1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

(B)

Fig. 7-11. Dual-range converter, ÷10 mode

8

COUNTER READOUT CIRCUITS

After a counter has ended a count cycle it often becomes necessary to know the value of the last count made. Readout Circuitry is designed to perform this function. Direct readout from a counter is in the less convenient binary form. Readout circuitry usually decodes the information from binary to the more convenient decimal form.

Nixie tubes

The value of the count is often presented visually by means of Nixie tubes. The Nixie is a type of gas-filled tube with 10 cathodes and 1 anode. Each cathode is shaped to form a digit. By applying anode voltage and grounding one of the cathodes the gas is ionized. In ionized gas tubes, the cathode is surrounded by glowing gas. The glow around the specially shaped cathode is seen by a viewer as a digit.

Readout circuitry to drive a Nixie tube must have ten output lines. Each line connects to one of the Nixie cathodes. By grounding one line at a time the digital value of the count stored is made visible.

The count value is often required for external recording devices such as tape printers or card punchers. These devices accept binary digital information. For this reason readout circuits also connect to external connectors making the count (in binary form) available.

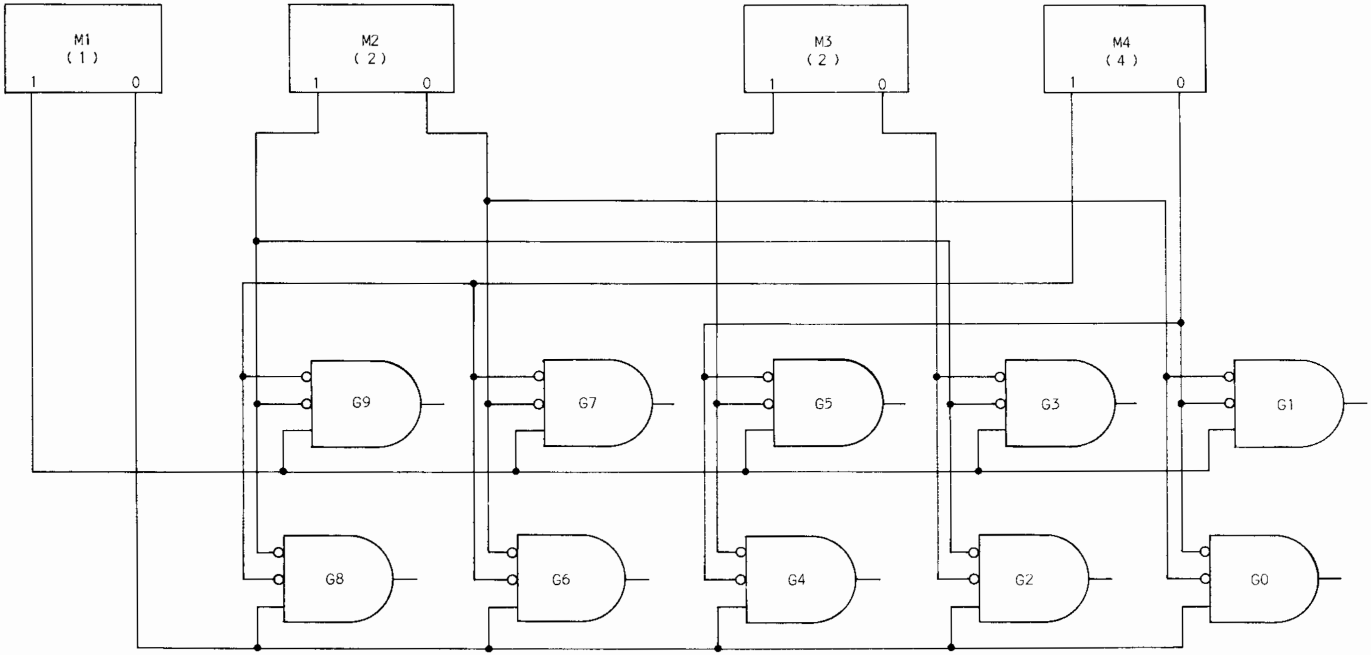


Fig. 8-1. Ten-line readout circuit.

Examples of decoding and nondecoding readout circuits are included in this chapter.

4,2,2',1 binary-to-decimal logic

Fig. 8-1 shows a binary counter with 4, 2, 2', 1 logic and a group of AND gates that convert the binary number contained in the counter to a decimal output. The output consists of 10 lines representing the decimal number 0-9. For any decimal number one of the ten lines will have a 1 output. All others will be 0.

The AND gates require 1's for each of their three inputs before their output will be a 1. An AND gate output of 1 indicates its assigned number is in the counter. Suppose the counter contained a 0. All 1 outputs would be 0's and all 0 outputs would be 1's. If the 1 output of M1 is a 1, the number contained in the counter is an odd number (1,3,5,7,9). If the 1 output of M1 is a 0, the counter contains an even number. In the example 0 is an even number and the 0 output of M1 is a 1. This enables Gates G8, G6, G4, G2 and G0. One of G8's inputs is connected to the 1 output of FF2 which is a 0. G8's output cannot be a 1. One of G6's inputs is connected to the 1 output of FF4 which is a 0. G6's output cannot be a 1. G4 and G2 also have a 0 input (G4 from M3 and G2 from M2). Neither of these gates can have an output of 1.

G0 has a 1 input from M1, M2 and M4. G0 and only G0 has an output of 1 indicating the counter contains a zero.

Suppose the counter contained the number 7 (1011). M1 would indicate an odd number (its 1 output is a 1). G7 would have a 1 input from M1, M2 and M4. G7 and only G7 would have an output of 1.

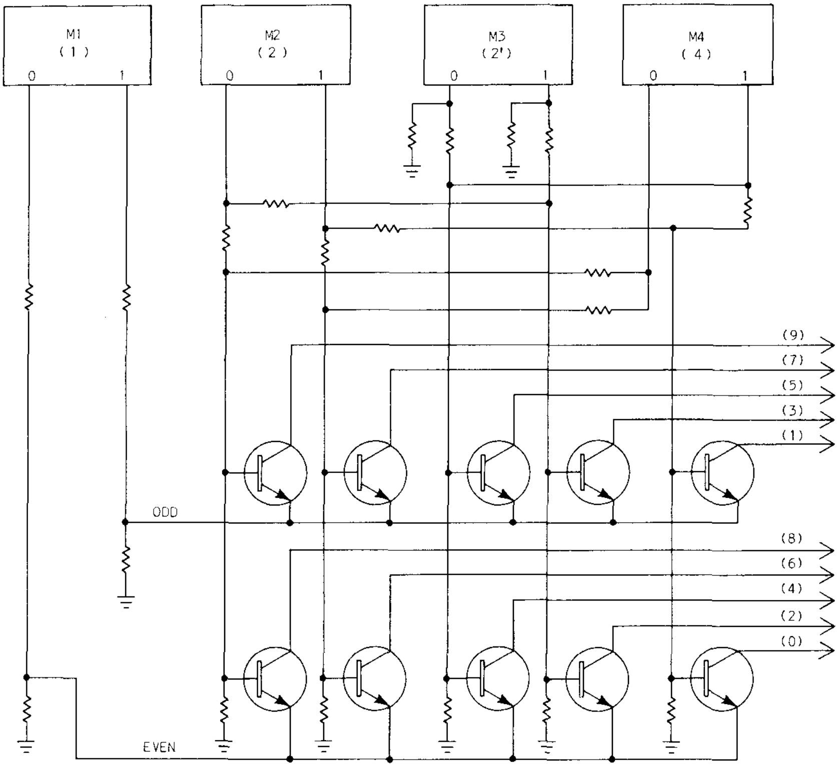


Fig. 8-2. 4, 2, 2', 1 binary-to-decimal converter.

4,2,2',1 binary-to-decimal circuitry

Fig. 8-2 shows the circuitry for 4, 2, 2', 1 code binary-to-decimal conversion. Each transistor has its emitter connected to either the 1 or 0 output of M1. These lines will enable either all the even or all the odd numbered transistors. The base of each transistor is connected to the output of two other FF's through resistors (see Fig. 8-3). Q9, for example is connected to the 0 output of M2 and the 1 output of M3. In order for the transistor to conduct both base inputs must be high *and* the emitter must be low. With only one base input high the transistor will conduct but not enough to pull its collector to 0 V. 0 V at the collector indicates a true condition.

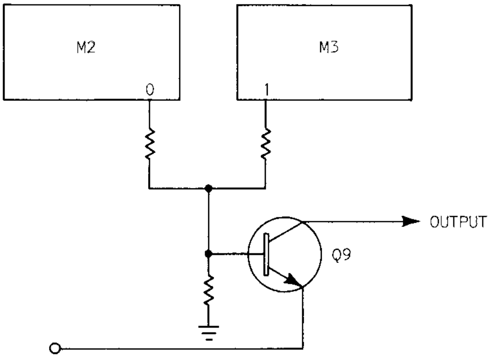


Fig. 8-3. Partial readout circuit.

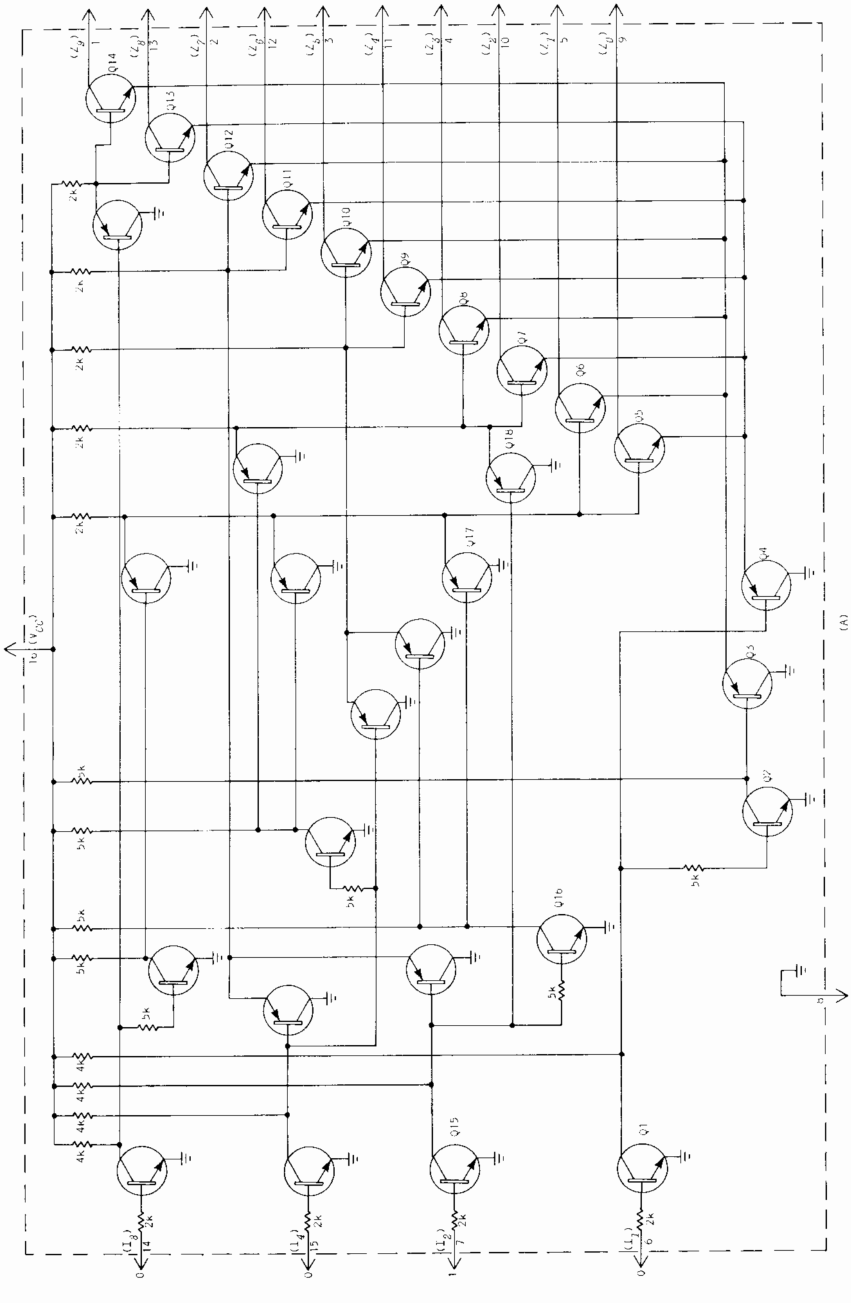
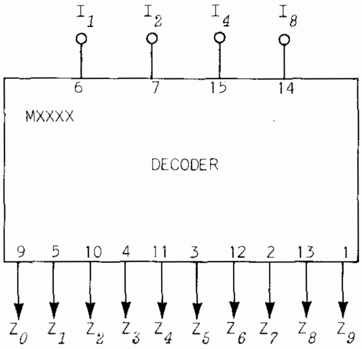


Fig. 8-4. Fairchild Type 960.

Fairchild 960

The Fairchild Micrologic 960 decimal decoder driver is housed in an epoxy dual in-line flat pack with sixteen leads. A schematic of the device is shown in Fig. 8-4A. This device decodes a binary-coded decimal (BCD) input to a ten line output. The output of the chip drives the numeral cathodes of a Nixie indicator tube. There are ten outputs labeled Z_0 to Z_9. The Z_0 would connect to the zero cathode, Z_1 to the one cathode, and so forth. The inputs accept all sixteen binary combinations but only BCD is permitted.

The chip contains thirty transistors and twenty-one resistors. The logic diagram symbol for the chip is shown in Fig. 8-4B. A decoding truth table is shown in Fig. 8-4C.



(B)

I_1	I_2	I_4	I_8	Z_0	Z_1	Z_2	Z_3	Z_4	Z_5	Z_6	Z_7	Z_8	Z_9
0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	0
0	0	1	1	0	0	0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0	0	1	0	0	0	0
0	1	1	0	0	0	0	0	0	0	1	0	0	0
0	1	1	1	0	0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	0	0	0	0	1

(C)

Fig. 8-4. Fairchild Type 960.

As an example of decoding, consider BCD number 0010 (2_{10}). The inputs are: I_1, I_2, I_4, and I_8. In BCD form, four binary digits are presented in a 1, 2, 4, 8 arrangement. The 8 bit connects to I_8 and so forth. See Fig. 8-4A. The zero on the I_1 input turns Q1 on. The collector pulls down to ground. Looking at Q2 we see that Q2 goes off. As this collector pulls high, emitter-follower Q3 pulls the emitters of output driver transistors Q6, Q8, Q10, Q12 and Q14 high, making it impossible for these transistors to turn on. Note that I_1 bit determines whether the final output line shall be odd or even. The transistors which the I_1 bit turned off are the transistors which drive odd number outputs Z_1, Z_3, Z_5, Z_7, and Z_9.

Return to the output of Q1. We see that the ground at the collector of Q1 also couples to the base of Q4. Q4, a PNP transistor with its base low, acts as emitter follower. Q4 emitter pulls the emitters of transistors Q5, Q7, Q9, Q11 and Q13 low, enabling them to turn on. The base of only one of these transistors is high. Look at the I_2 input and at transistor Q15. Notice that the I_2 input is low or true at this time. Q15 is off. Q15 collector rises. This high couples to the base of transistor Q16. Q16 turns on. The collector of Q16 drops pulling the base of emitter follower Q17 low. Q17 pulls the bases of Q5 and Q6 low. Therefore, neither of these two transistors can be on. The collector of Q15 also connects to Q18 base. The high at Q18 base turns it off. Q18 emitter, returned to V_{cc}, pulls the base of Q7 high, Q7 turns on. The base of Q8 is also pulled high. Q8 emitter is pulled high by the input drive from Q4. With equal potentials on emitter and base, Q8 cannot turn on. In a similar manner a check all the way up the row of transistors reveals that all other transistors are off with this particular input combination. For any other BCD input from decimal zero to decimal nine, one and only one of the output transistors, Q5 through Q14, will be on.

storage register

Some types of counting devices operate in an alternating mode. The circuit first counts and then displays the results. This method of operation is time consuming since the circuitry cannot make a new count until display time is finished. A different approach lets the counter make the count and then transfer the total of the count to a storage circuit. The storage circuit displays the results. Meanwhile the counting circuits are free to make a new measurement. In this manner, considerable time can be saved. The storage circuit is called a storage *register*.

Fairchild 923

Storage registers can be formed using JK FF's such as the Fairchild Type 923. See Fig. 8-5. The JK FF is operated on a clocked mode. The P and C_p inputs are connected together. The P input of a 923 responds only to a positive level input while C_p is sensitive only to a negative edge. This circuit is capable of storing one bit of the readout information. If 12 bits are to be read out of a counting device, 12 storage register circuits of this type are required. A signal appearing after the counting circuits have completed their measurement must be supplied. This signal is called Register $\overline{\text{Set}}$.

The bar over the word Set means that this waveform will go from true to false when activated. The bit to be stored couples to the K input of M1. Output is taken from the 0 output (pin 5).

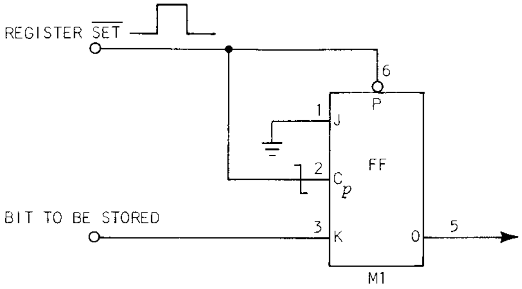


Fig. 8-5. Storage register.

Assume that the bit to be stored is a 0 (high level), in this case approximately +3.8 V. From discussion of the Fairchild Type 923 FF* recall that a positive level presets the FF. Pin 1 goes true, pin 5 goes false. With the appearance of Register $\overline{\text{Set}}$ the FF Presets. The 0 output goes false. When the negative edge of Register $\overline{\text{Set}}$ appears M1 clocks. With K held false, clocking M1 can only make the zero output go high. (For this example no change occurs.) M1 remains with the zero output false until the next Register $\overline{\text{Set}}$. The circuit can be said to store the zero bit which was fed in.

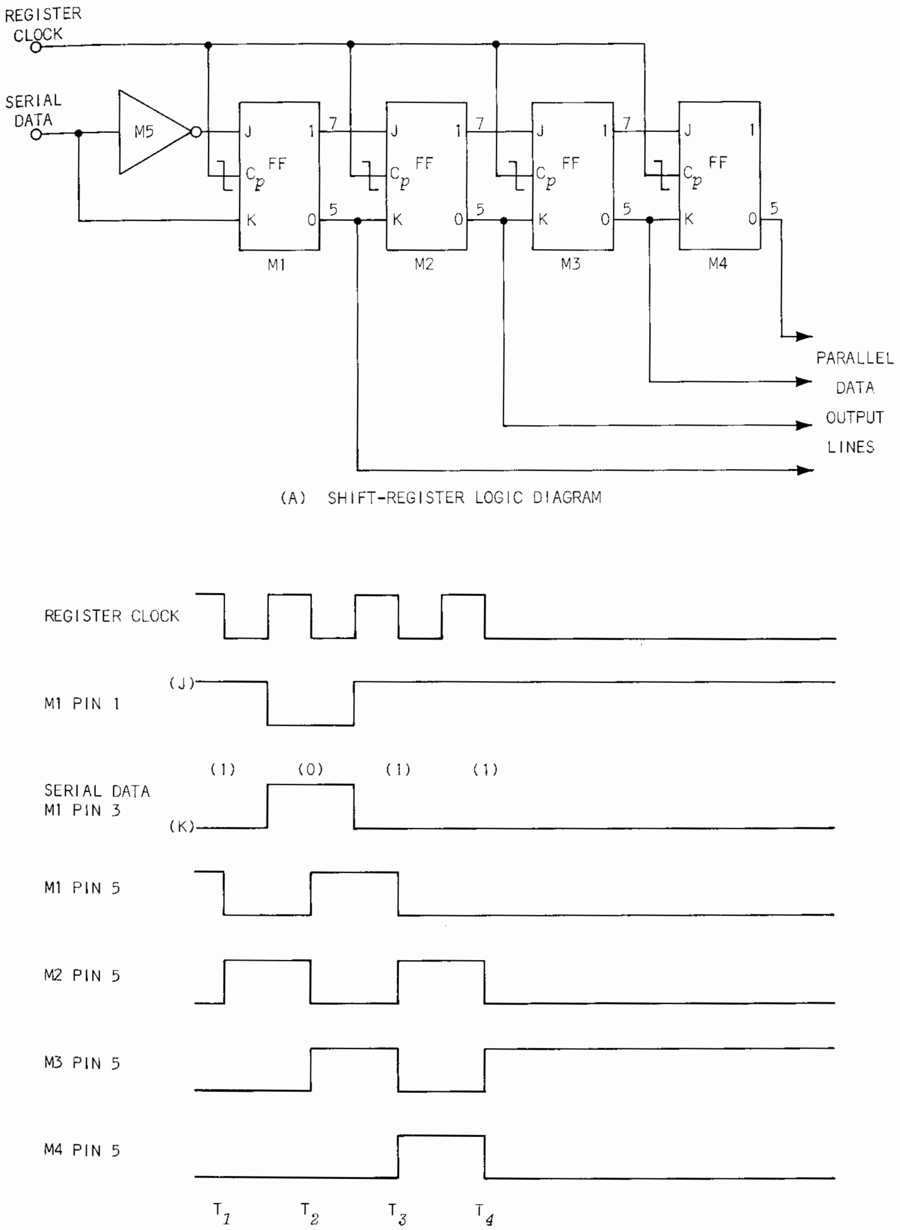
Now assume that the bit to be stored is a true level. When the positive edge of Register $\overline{\text{Set}}$ appears, pin 5 goes to 0. With the negative edge of Register $\overline{\text{Set}}$ M1 toggles because both J and K inputs are true. Pin 5 which was at 0 goes to 1. The circuit has stored the one bit fed in.

Register circuits are used for purposes other than simple storage. Data is often presented as a series of serial bits on a single line. An oscilloscope monitoring a line feeding in such a series of bits would show a set of true and false pulse waveforms. Usually these serial bits must be converted to parallel bits.

For example in the Tektronix Type 240, a serial-to-parallel conversion operation is performed which takes groups of four bits in serial form and converts them to four bits in parallel form. Each group of four Serial bits in are coupled out via four parallel lines, one bit per line. A Shift Register which performs this function is shown in Fig. 8-6A.

A Clock signal is always required by a Shift Register. The Register Clock signal informs the circuit when a bit is being applied to the Register. The signal also serves to shift data through the Register.

*See Chapter 6 page 79.



(A) SHIFT-REGISTER LOGIC DIAGRAM

(B) SHIFT-REGISTER LADDER DIAGRAM

Fig. 8-6. Four-bit shift register.

An observer using a test oscilloscope connected to the serial data line would find it difficult to detect the separation between the two adjacent 1 (or 0) bits. This demonstrates the added value of the clock pulse. We define the status of Serial Data line by stating: Serial data is a 1 or a 0 depending upon its exact level at the time the negative edge of Register Clock occurs.

These actions are most easily seen by tracing the operation cycle of the circuit for consecutive serial data bits. See Fig. 8-6B. Assume that the first four data bits into the register input are 1011.

The serial data is connected to the input of inverter M5 and to K of M1. The inverted output of M5 connects to J of M1. J and K always have opposite level inputs.

At T_1 serial data is a 1. This places M1 pin 1 at 0 and pin 3 at 1. With the negative edge of clock all four FF's switch simultaneously.

The reader should recall that no circuit can change states instantaneously. This means that the switching of each FF is determined by the J and K input levels that existed *prior* to each clock pulse.

Thus, M1 had pin 5 at 0. This 0 transfers into M2 because the high at M2 pin 3 causes M2 pin 5 to go high.

At T_2 serial data is a 0. With the clock at T_2 M1 pin 5 goes to 0. M2 pin 5 goes to 1. M3 pin 5 goes false. M4 pin 5 does not change.

At T_3 serial data is a 1. M1 pin 5 remains a 1. M2 pin 5 goes false. M3 pin 5 goes true. M4 pin 5 goes false.

At T_4 serial data is a 1, M1 pin 5 remains a 1. M2 pin 5 goes to 1. M3 pin 5 goes to 0. M4 pin 5 goes to 1.

After T_4 the status of the zero output lines of the four FF's *read from bottom to top* is 1 0 1 1. These bits are now available on four parallel lines. External circuitry capable of accepting parallel input data may now be signaled to accept this data. A typical external circuit would be another kind of shift register with four parallel inputs.

Note that if a device was connected to M4 pin 5 the data could be read out of the shift register in serial form. Therefore, a shift register typically may have either parallel or serial outputs or both.

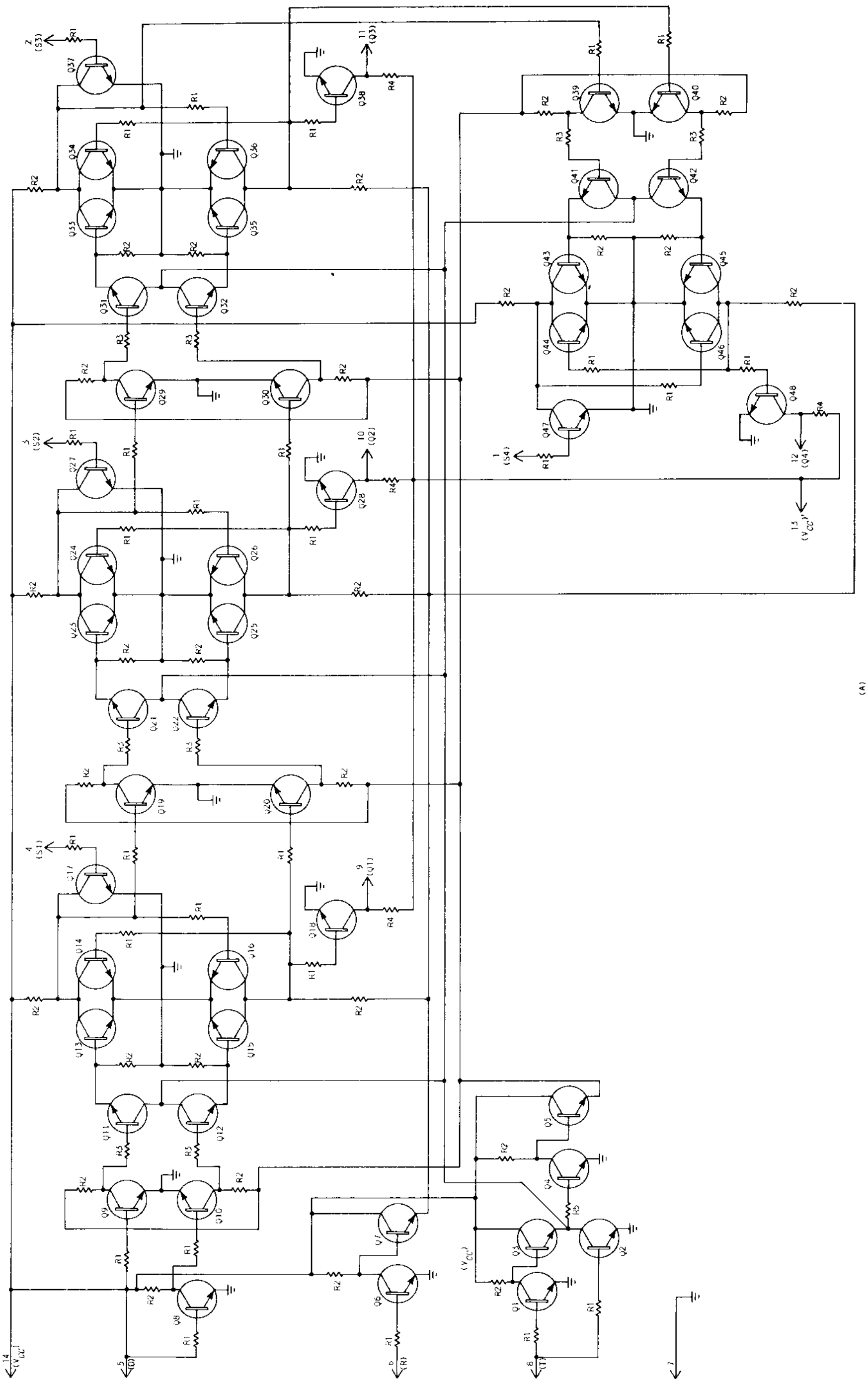
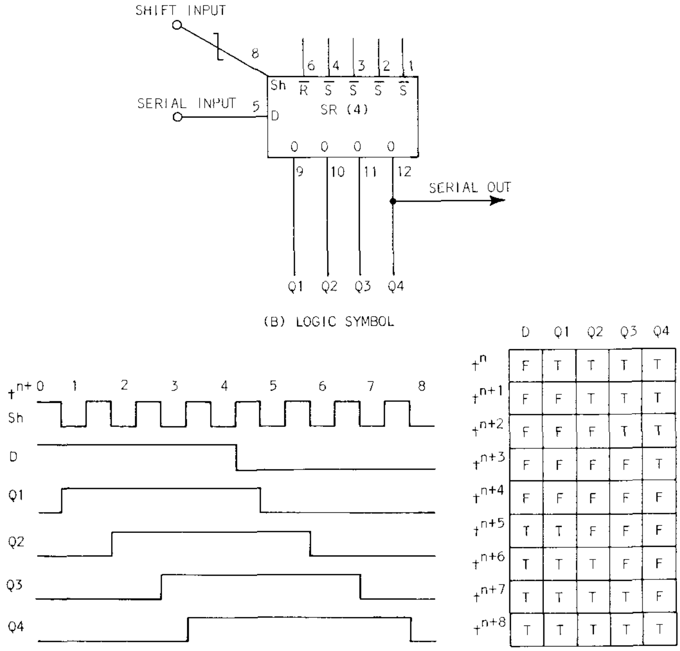


Fig. 8-7. Fairchild 997 shift register.

Fairchild 997

A four-bit shift register which is incorporated on one IC chip is shown in Fig. 8-7A. This is a Fairchild 997. It is housed in a dual in-line flat pack with 16 leads. The chip has 48 transistors. It is intended by the manufacturer to serve as a multipurpose shift register. The chip accepts either serial or parallel entry and provides either serial or parallel output. In Tektronix instruments, such as the Type 240, this device is wired so that parallel entry is denied. The device uses serial entry and both serial and parallel output.

Fig. 8-7B shows a logic symbol for the 997. Pins 9, 10, 11 and 12 are the parallel readout pins, pin 12 is also the serial output. The Right Shift input serves the same purpose as the clock input in the example of Fig. 8-6. Pin 8 is the shift input which is negative-edge sensitive. Pins 1, 2, 3, and 4 are Set inputs wired together. Each of these inputs connects to one of the FF's within the schematic diagram.



SHIFT INPUT
8
6
4
3
2
1
Sh
$\overline{R}$ $\overline{S}$ $\overline{S}$ $\overline{S}$ $\overline{S}$
SERIAL INPUT
5
D
SR (4)
0 0 0 0
9
10
11
12
SERIAL OUT
Q1 Q2 Q3 Q4
t^{n+0} 1 2 3 4 5 6 7 8
Sh
D
Q1
Q2
Q3
Q4

(B) LOGIC SYMBOL

	D	Q1	Q2	Q3	Q4
t^n	F	T	T	T	T
t^{n+1}	F	F	T	T	T
t^{n+2}	F	F	F	T	T
t^{n+3}	F	F	F	F	T
t^{n+4}	F	F	F	F	F
t^{n+5}	T	T	F	F	F
t^{n+6}	T	T	T	F	F
t^{n+7}	T	T	T	T	F
t^{n+8}	T	T	T	T	T

(C) WAVEFORMS AND TRUTH TABLE

Fig. 8-7. Fairchild 997 shift register.

right shift

The term right shift describes the direction a bit will move through the shift register. On the schematic when the Right Shift input is activated, the pulse moves from left to right. Within certain Tektronix instruments a reference is made to a Left Shift register. In this case while the schematic would be drawn in a similar manner to Fig. 8-7A, the wiring would be reversed so that the signal literally moved from right to left on the page.

left shift

A truth table and waveform timing diagram appear in Fig. 8-7C. The reader may examine the sequence of operation with the aid of the truth table and timing diagram.

The 997, in common with Fairchild micrologic 900 series IC's, is activated by a negative edge at the shift input. Data is applied to pin 5 in serial form and processes through the shift register with each negative edge of the shift pulse train.

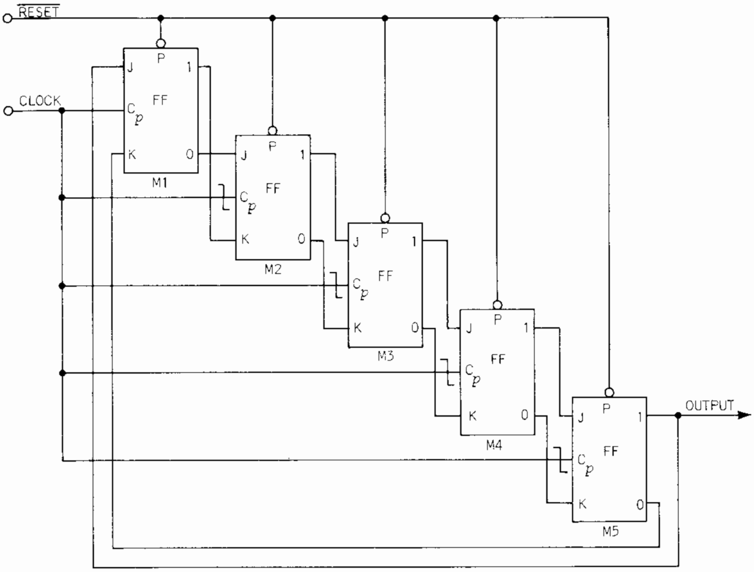
÷5 ring counter

A special circuit which combines the features of a shift register with those of a counting circuit is shown in Fig. 8-8A. The circuit is designed to perform a divide-by-5 function. For every five clock pulses coming into the left side of the diagram, one pulse leaves the output. This circuit is periodically reset by generating a Reset pulse.

A truth table for the operation of the circuit is shown in Fig. 8-8B and a ladder diagram in Fig. 8-8C. For FF's M2, M3 and M4 the one and zero outputs are connected to J and K inputs of the next FF respectively. However, M5 to M1, and M1 to M2, the two outputs cross over. That is, 1 goes to K and 0 goes to J.

At time T_0 the Reset pulse occurs. All FF's set to one. The 1 output waveforms of each FF are as shown.

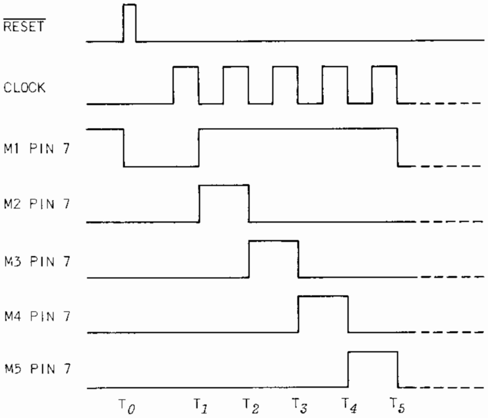
With the negative edge of clock (T_1) M1 and M2 toggle. M1 toggles because J is false and K is true. The 1 output of M1 goes to 0 and the 0 output goes to 1. M2 toggles for the same reason. M3 and M4 do not change.



(A)

TIME	M1		M2		M3		M4		M5	
	1	0	1	0	1	0	1	0	1	0
T_0	1	0	1	C	1	0	1	0	1	0
T_1	0	1	0	1	1	0	1	0	1	0
T_2	0	1	1	0	0	1	1	0	1	0
T_3	0	1	1	0	1	0	0	1	1	0
T_4	0	1	1	0	1	0	1	0	0	1
T_5	1	0	1	0	1	0	1	0	1	0

(B)



(C)

Fig. 8-8. ÷5 ring counter.

At T_2, M1 remains the same, M2 switches with J low and and K high. M3 reverses states, M4 and M5 remain the same.

At T_3, M1 remains the same, M2 remains the same, M3 reverses states, M4 reverses states, M5 remains the same.

The reader may continue through the sequence by utilizing both the truth table and the waveform diagram and he will find that at the lead labeled *Output*, one complete pulse appears between T_4 and T_5. At T_4 Output goes false, at T_5 Output goes true again. This generates one negative edge at the output for five negative edges on the clock pulse input. Hence, the divide-by-five function has been performed.

This type of counter is frequently called a ring counter. The pulse which the reader will notice at the output of M2 and M3 and M4 is circulated around the loop (or ring) of the circuit. This circuit performs a continuous divide-by-five function whenever clock pulses appear.

INDEX